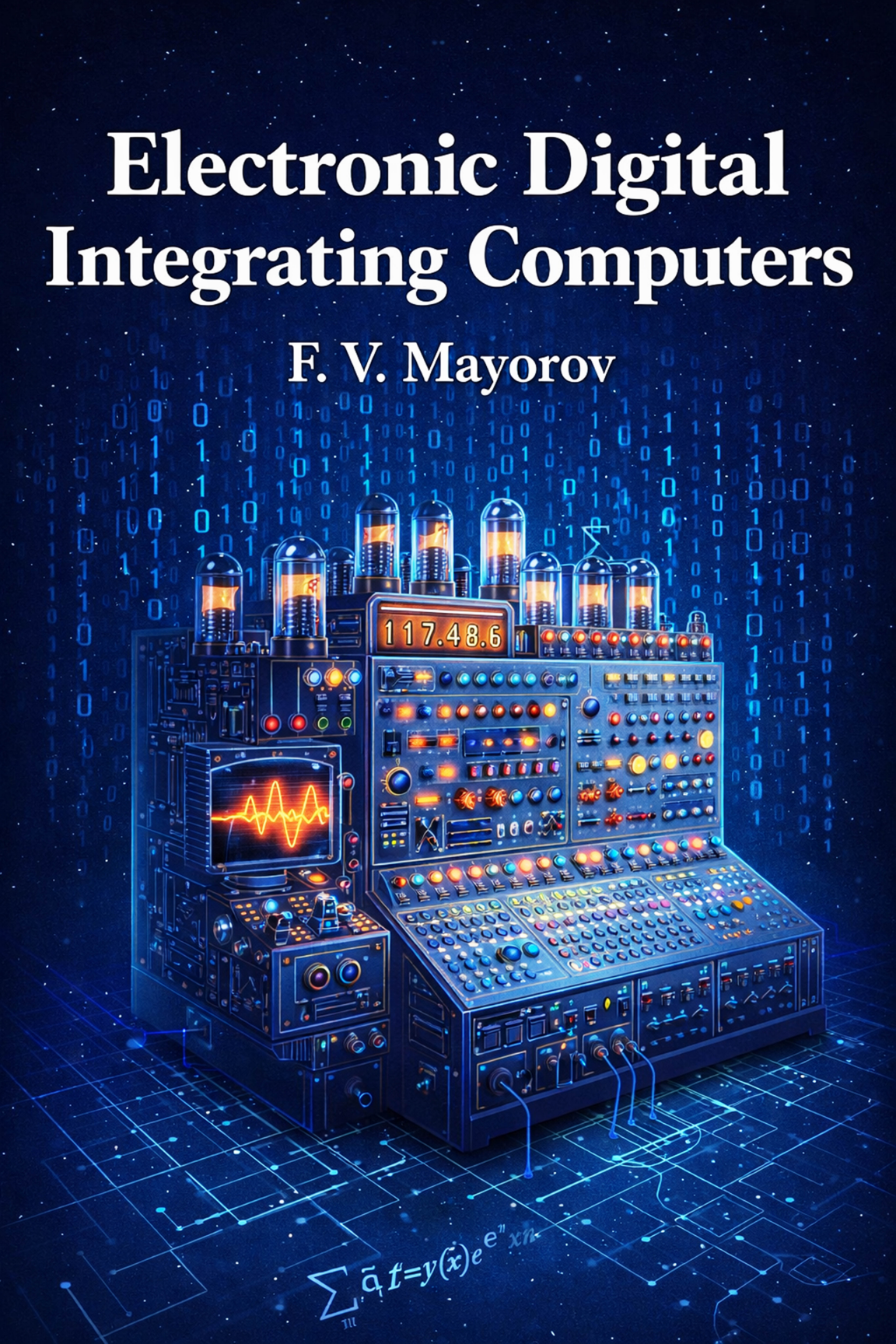
Electronic Digital Integrating Computers

F. V. Mayorov



ELECTRONIC DIGITAL INTEGRATING COMPUTERS

Digital Differential Analyzers

ELECTRONIC DIGITAL INTEGRATING COMPUTERS

Digital Differential Analyzers

F. V. MAYOROV

translated by Scripta Technica Inc

editor Dr YAOHAN CHU
Data System Center
Radio Corporation of America

LONDON ILIFFE BOOKS LTD
NEW YORK AMERICAN ELSEVIER
PUBLISHING COMPANY INC.

Originally published in the U.S.S.R. under the title
Elektronnye tsifrovye integriruyushchie mashiny
by Mashgiz, Moscow, 1962

English edition first published in 1964 by
ILIFFE BOOKS LTD, Dorset House, Stamford Street
London, S.E.1

© Iliffe Books Ltd, 1964

Published in the U.S.A. by
American Elsevier Publishing Company Inc.
52 Vanderbilt Avenue, New York, N.Y. 10017

Library of Congress Catalog Card Number 64-25379

Printed and bound in England by
Butler and Tanner Ltd
Frome, Somerset
BKS 5036

Editor's Preface

This book presents the principles, organization, applications, circuits, memories, and input-output devices of electronic digital differential analyzers (also called digital integrating computers). It describes both serial and parallel machines as well as both binary and decimal types. Although it has been more than a decade since the appearance of the first digital differential anlayzer in the United States, Mayorov's book is the first one, as far as the editor knows, devoted entirely to this subject. For this reason, this book serves as a useful volume to those who are interested in this subject.

The material contained in this book has been drawn mostly from the USA and the USSR. Since there have been many papers on this subject in the United States and in the United Kingdom that the author has not included in the bibliography, a supplementary bibliography is provided for those who wish to pursue this subject further.

The popularity of the digital differential analyzer has been overshadowed by rapid and wide introduction of the general-purpose digital computer because of the latter's flexibility for many applications, including nonnumerical applications. However, interest has recently been generated in the USA in the iterative type of digital computer for special applications. The principles of digital differential analyzers may to some extent be applicable to such machines. Consequently, the publication of this English version of Mayorov's book seems to be timely.

Chevy Chase, Md. Yaohan Chu

Contents

INTRODUCTION ix

CHAPTER I. Principles of Digital Integration 1
1. Types of Digital Integrating Computers (DDA) 1
2. Principles of Integration 7
3. Trapezoidal Integration 12
4. Binary Coding 14
5. The Integration Operation 19
6. Ternary Method of Coding Increments 23
7. Serial Integrators Using A Magnetic Drum Memory . . 31
8. Interconnection of Integrators 38
9. Insertion of Initial Conditions. Problem Switching . . . 41
10. Servo Integrator 44
11. Scale Factors 49

CHAPTER II. Methods of Solution of Mathematical and Logical Problems 54
1. Solution of Differential Equations 54
2. Basic Algebraic Operations 57
3. Generation of Algebraic, Trigonometric and Other Functions. Transformation of Coordinates. 60
4. Solution of Linear First-Order Algebraic Equations . 76
5. Solution of Higher-Order Algebraic Equations 78
6. Solution of Differential Equations With Given Boundary (Initial and Final) Conditions 80
7. The Differentiation Process 82
8. Determination of Extreme Values 86
9. Limiting Operation 88
10. Logical "Sign" Operation 91
11. Solution of Integral Equations 95
12. Correlation Analysis 98
13. Program Preparation 100

CHAPTER III. Digital Computer Circuits 102
1. Switching Operation of the Junction Transistor 102
2. Inverter Amplifiers 107
3. Emitter Followers 109
4. Static Transistor Flipflops 115
5. Unsymmetrical Static Flipflops With Emitter Coupling 124
6. Standard Elements of Transistorized Digital Computers 126
7. High-Speed Current Switches 128
8. Voltage Limiters and Regulators 136

9. Diode and Diode-Transformer Coincidence and Inhibit Circuits 140
10. Dynamic Flipflops 147
11. High-Power Dynamic Flipflops 153
12. High-Power Pulse Generators and Shapers 155
13. Dynamic Flipflop Shift Registers 157
14. Counting Circuits 159
15. Diode-Transformer Decoders 163
16. Serial Binary Adders 166

CHAPTER IV. Storage Devices 172
1. Magnetic Drum Storage 172
2. Read-Write Amplifiers for Magnetic-Drum Storage . . 174
3. Magnetic Internal Memory (MIM) With Ferrite Plates or Cores 176
4. Magnetic Internal Memory 177
5. Address Decoding 183
6. Read and Write Operations 187
7. Drivers 191
8. Diode-Capacitor Registers 195
9. Write Amplifier 197
10. Read Amplifier 198
11. Diode-Capacitor Read Register 203
12. Memory Not Using Current-Coincidence Principle . . 205
13. Magnetostrictive Delay-Line Storage 206
14. Acoustic Delay-Line Storage 213

CHAPTER V. A Serial Binary Digital Integrating Computer . . . 220
1. Block Diagram 220
2. General Characteristics of Computer 223
3. Integrating Unit 224
4. Control Unit 231
5. Program Code Generation Unit 233
6. Increment Input Unit 236
7. Output Converter for Motor Control 238
8. Increment-Pulse Storage 240
9. Writing and Reading of Increment Pulses 242
10. Storage of Input Channel Data in the MIM 251
11. Digital Integrating Computer Using Ultrasonic Delay Lines 252
12. Initial Condition Register 256
13. Double Integrator 257

CHAPTER VI. A Serial Decimal Digital Integrating Computer . . 259
1. Decimal DDA With A Magnetic Drum 259
2. 5211-Code Decimal Adder 264
3. Excess-3-Code Decimal Adder 268
4. Increment Sum $\Sigma_{\Delta y}$ for A Decimal Adder 272

CHAPTER VII. Multichannel Analog-to-Digital Converters 275
1. A Converter Using Diode Switches 276

2. A Converter Using Transistor Switches 281
3. A Digital Potentiometer 284
4. A Multichannel Converter Using A Digital Potentiometer 286
5. A Multichannel Transistor Switch 295
6. A Shaft-Position Converter 297

CHAPTER VIII. Input-Output Devices 301
1. Punched Tape Device 301
2. Input Devices for Graphical Data 308
3. Graphical Plotting of Computed Results 313
4. High-Speed Function Generators 314

CHAPTER IX. A High-Speed Parallel Digital Integrating Computer 324
1. Block Diagram of A Parallel Integrating Computer . . 324
2. Delay-Line Storage Register 326
3. Digital Integrator 329
4. Multiplier Integrator 334
5. Pulse Increment Adder 339
6. Digital Servo 341
7. Insertion of Initial Data 344

CHAPTER X. Digital Integrating Computer Applications 348
1. Automatic Control of Production Processes 348
2. Digital Programmed Control of Machine Tools 353
3. DDA Control of Machine Tools 359
4. Combined Use of DDA and DAC 365
5. Analog Applications of the DDA 369

BIBLIOGRAPHY 371
SUPPLEMENTARY BIBLIOGRAPHY 373

Introduction

In recent years a new type of computer, based on the principle of digital integration, has found increasingly wider application. Such computers, combining the advantages of digital and analog machines, were first referred to as digital differential analyzers (DDA).

Further development of digital differential analyzers evolved a class of "incremental computers" based on the principle of summation of increments. We shall refer to machines of this class as digital integrating computers (DIC) since the result of calculation therewith is obtained by step-by-step accumulation of increments.

There are two types of digital integrating computers:

a) Integrating computers with a fixed increment, including digital differential analyzers of both serial and parallel types;

b) Integrating computers with a variable increment wherein the arithmetic operations as well as the integration operation may take increments of 2^0, 2^1, 2^2, 2^3, 2^4, etc. In such a computer the iteration process is achieved automatically with a variable increment (larger increments at first and then smaller) without loss of accuracy.

Digital integrating computers with variable increment occupy a position between the conventional general-purpose digital computer (or digital arithmetic omputer DAC) and the DDA. In the former, calculations are made on numbers in full-length (i.e., with maximum possible increment). In the DDA, the increment is minimum and equal to unity.

Solution of mathematical problems by the integration method is now a distinct trend in the development of digital computers which has led to the development of a whole new series of special-purpose computers with a high efficiency and a high operating speed. General-purpose digital computers played the principal role in the first stage of computer development. It was subsequently seen that a special-purpose computer is simpler, more effective and faster in solving a specific class of problems than a general-purpose computer. General-purpose computers can no longer meet the extremely severe requirements in operating speed and computing rate in real-time control systems. They are complex and bulky. They require considerable time for manufacture and setup. They do not provide the necessary reliability and readiness for operation at any time, which is particularly important for real-time control applications. The large number of electronic components (as many as tens of thousands) and, in particular, the difficulty of setting up the computer accounts for the difficulty of their production in quantity.

Operation of a general-purpose computer involves considerable expense. Its unreliability necessitates almost daily testing and preventive maintenance, requiring a large staff of skilled personnel. The time spent in setting up and programming of a problem may amount to weeks or even months.

The use of semiconductor components instead of vacuum tubes has increased the operational reliability of general-purpose computers, but the extremely large number of these components does not give the required reliability.

For most practical applications where the problem is solved in accordance with a previously prepared program, a general-purpose computer is not necessary. In fact, in terms of the stated problem, such a computer is unnecessarily complex and relatively inefficient.

As was pointed out at a conference on computer engineering in 1956, "Large computers should be built only for large computing centers in which effective use of such computers is possible."

Thus, technical and economic factors dictate the use of simpler, more reliable, economical, and compact special-purpose digital computers which may be produced in quantity for use in many applications.

The need for simple, compact digital computers suitable for solving differential equations led to the development of digital differential analyzers in 1949. The first DDA was built in the United States in January of 1950 (i.e., approximately five years after the advent of the first electronic digital computer, ENIAC, in 1945). Since 1950, digital differential analyzers came to be used primarily for scientific and technical calculations associated with solution of systems of differential equations (e.g., for solving problems in aerodynamics and ballistics as well as problems associated with research and development of missiles and satellites). The DDA has recently been applied to automatic control systems.

In the USA, England, West Germany, Japan, Australia, and other countries, it has been reported in the press that the development of digital differential analyzers in the past ten years has brought out a new class of digital computers. The importance of this trend in digital computing can be seen from various publications. For example, in the USA a National Council on Digital Differential Analyzers has been created to coordinate research and development in this field.

A number of the largest aircraft companies and electronics corporations are developing and producing digital differential analyzers (North American, Bendix, Packard-Bell, Litton, Stevens, General Electric, et al.). More than twelve different types of DDA have been developed. For example, Bendix Aviation developed three types of DDA in four years: D-12 (1954), D15D (1956) and, finally, D-18 (1958). North American Aviation has also developed several DDA: CRC-105, NATDAN, VERDAN and the MACE aircraft computer (flight tested in 1956). It has been reported in the press that Bendix DDA's of types D-12 and D15D have been installed on naval vessels for navigation computation and a DDA constructed

by the Computer Research Corporation has been installed on artillery ranges.

The Arma Corporation has developed a miniaturized DDA for use in an intercontinental ballistic control system presently under development by Arma. This DDA solves the control problem in an inertial guidance system.

Packard-Bell has constructed an ultra-high-speed digital differential analyzer TRICE (transistorized) for use in real time control systems, for analog simulation, and for scientific-engineering computations. The machine is capable of 100,000 problem solutions per second with a word length of 26 binary digits. Single solution of a complex problem is done in 10 μsec, whereas a large-scale high-speed general-purpose computer requires 24 times as much time (that is, 240 μsec) for the same solution and in 10 μsec is barely able to perform the multiplication of two numbers.

The ultra-high-speed digital differential analyzer has been used in computing the trajectories of ballistic missiles. In one second the machine solves an entire system of equations of motion of a missile for a distance of 5,000 miles with an increment of 1/20 of a mile. The error in trajectory computation does not exceed 1/25,000 or 1/5 mile. Such computing speed cannot be achieved in any of the existing general-purpose digital computers. In order to be capable of having such a computing speed, a general-purpose digital computer would perform several million integrations per second.

A similar solid-state DDA is under construction by the Stevens Institute. It is reported that the analyzer will provide 8,192 solutions of a certain system of differential equations per second and will be used for computing trajectories and point of interception of moving objects.

In the last few years Litton Industries has constructed several types of compact DDA. The first was the DDA-20 (with 20 integrators) of the desk type and was designed for solving problems in the kinetics of chemical reaction. Litton has subsequently constructed several DDA models for aircraft and missiles.

Recent advances have led to newer types of digital integrating computer with variable increment including the digital incremental computers developed by General Electric: GEVIC (with variable increment), CVIC (with conditionally variable increment), et al. These compact computers, which are designed for missiles, fighters, and bombers, are also used for control of Polaris missiles in submarines and for inertial navigation of the missiles themselves. These computers weigh 45 pounds and occupy 0.8 ft . The variable increment contributes to the high dynamic quality in automatic control systems. The arithmetic and logic operations as well as integration are performed at extremely high speed. In addition to the above mentioned digital integrating computers, special types of DDA have been developed for use in aircraft, missiles, and submarines. These include the VERDAN produced by North American Aviation in the USA, the CORSAIR in England, a DDA for rockets of types X-10 and 1100 in the USA, etc. In 1957 a DDA known as TEDDA was developed in Japan.

The simplicity and compactness of the DDA permits its successful use for control purposes. By means of a DDA various computations associated with the solution of algebraic or trigonometric equations can be performed. The computer can be used for programmed control and for execution of purely logic operations. As a control computer the DDA is capable of continuous tracking and multiple checking by a digital servomechanism and capable of pulse-counting, storage and comparison of given and computed quantities. Another important property of the control computer is its ability to switch automatically from the problem being solved to the control program, depending on the results of computation or on a change in an external parameter. For many control and computation functions, the DDA uses the same electronic equipment in high-speed, serial operation that permits minimization of the size of electronic control equipment.

A promising application for the control purpose is the combined use of DDA and DAC (digital arithmetic computer or general-purpose digital computer). In view of the requirements for modern control systems, the speed of DAC is completely inadequate for computation, particularly in the presence of a large amount of computation and of the high speed requirements due to dynamics of the controlled process. Even an extremely large general-purpose computer cannot handle the computations necessary for a real-time control of moving objects.

The combined use of a DAC and DDA or a variable increment integrating computer permits a high computing speed and control at sufficiently high rate.

An example of such a hybrid computer is that built by Librascope Inc. (USA) for underwater weapon control. It is a simple and relatively slow, serial arithmetic machine and two digital integrating computers. The serial arithmetic machine controls the entire system and inserts the initial data into the two digital integrating computers, which solve the ballistics equations and guide the missile to the target. Each integrating computer is capable of performing 100 problem solutions per second. In solving one guidance problem, the integrating computer performs 256 different operations.

A similar hybrid computer has been developed in Great Britain at the Farnborough Flight Test Center. When it was found that the general-purpose digital computer at the center was not capable of achieving the required computation speed, a DDA was developed. The DEXAN [21] aircraft navigation computer also uses a DDA.

It has now been established that for many purposes the DDA is more effective than a general-purpose digital computer or an analog computer. For example, a US computer mathematian, Rutishauser, stated at a conference that "digital computers and analog computers are extremely unwieldy in comparison with the DDA for shipboard use." He further compared the setup time for solving an integral equation on a general-purpose arithmetic machine and on a DDA. It was found that the programming and problem solution time on a

general-purpose digital computer was two weeks, whereas on the DDA the time was two hours.

He regards the DDA as considerably better for computation than the general-purpose computer in the following classes of problems: solution of integral equations; evaluation of integrals in the field of probability theory; solution of Bessel equations and Bessel functions; solution of systems of algebraic and differential equations; determination of boundary conditions; computation of Fourier integrals or Fourier series, etc.

Another specialist, Dr. Braun, asserted that in order to increase the effectiveness of an existing general-purpose digital computer it is necessary to combine the latter with a DDA, since the DDA is not only superior to a general-purpose computer in many applications but may also replace an analog computer in those cases where considerable accuracy is required. The DDA is an extremely effective machine in computing continuous functions. It is more effective than a general-purpose digital computer in solving those problems for which the DDA is designed.

Dr. Rubinov (USA) stated that "there are real advantages to be gained in connecting DDA to a high-speed digital computer since the integration operation on even a slow DDA is comparable with the integration speed on a high-speed digital computer because the DDA is better organized for integration. In all probability, digital differential analyzers may also replace analog computers. A DDA is also required where an analog computer cannot provide the required accuracy and when the solution time of a problem is long (e.g., in inertial guidance and navigation of a missile). With an increase in complexity and nonlinearity of the problem, the DDA becomes an even more effective machine because the amount of equipment does not increase in proportion to the complexity of the problem, the accuracy of a DDA does not decrease with an increase in complexity of the problem, and the number of integrators required in the DDA is less than the number of amplifiers in an analog computer.

In analyzing the reasons for wide use of general-purpose digital computers in the USA, Dr. Braun stated that in the first place the requirements of a problem were not clearly understood and formulated and the general-purpose computer has flexibility in meeting changes in requirements. In the second place, in many applications the cost of developing the computer was justified. However, the lower effectiveness in using a general-purpose computer will necessitate the use of a special computer for each class of problem.

An example of such a special-purpose computer is the SPADE computer for solving partial differential equations. In DINA, an earlier differential analyzer, a relatively slow magnetic drum storage is used. Nevertheless, the speed of DINA was about 100 times faster than that of a high-speed general-purpose computer. This comparison further confirms the advantage of speed and economy of a special-purpose computer.

The above remarks show that the latest serial DDA possess a high problem solution rate of about several thousand solutions

per second. This means that the solution time of a problem is several hundred microseconds; such a solution time cannot be achieved in a high-speed general-purpose arithmetic machine. Since even a relatively slow DDA with magnetic drum storage is successfully used for computation, the DDA becomes more powerful when it employs high-speed semiconductor components.

The use of a parallel DDA permits an extremely high operating speed of 100,000 problem solutions per second with existing components and in the near future the speed may even reach 300,000-400,000 solutions per second (at a clock frequency of 10 Mc). The development of a high-speed DDA (having only one serial adder) presents no difficulty and requires an extremely small number of high-speed components, whereas the development of a high-speed general-purpose word-organized digital computer leads to great complication in circuitry and to an increase in component count. In fact, it was reported that the decrease of addition time in the latter is achieved by using considerably more electronic components and that the processing of data and instruction repertoire in such a computer becomes complicated. The American design specification for the high-speed general-purpose computer STRETCH called for two computers: a serial one for handling instructions and a parallel one for computation. The addition operation in the serial machine will be slower than in the parallel machine by a factor of only 3 or 4. This means that the parallel principle of arithmetic operations will no longer yield the same great advantage in speed as before.

A substantial disadvantage of general-purpose computers is the complex and lengthy task of programming the problem, whereas in a DDA it is not necessary to program the problem and the problem can be quickly solved.

The high computing speed and operating efficiency of a digital integrating computer result from the facts that:

a) In integration, the computer operates with increments of input quantities and not with the quantities themselves as in the arithmetic machine. This permits considerable increase in computing speed and in switching (or interconnection) of the integrators.

b) By using integration as a basic operation, operations of multiplication, division, extraction of a root, sine of a number, logarithm calculation, take a time equivalent to that for 2-3 addition operations (that is, much less time than in an arithmetic machine). Insertion of various experimental and tabular values into a digital integrating computer can be simpler than in an arithmetic machine and requires a memory of amaller capacity.

c) The memory access time in a digital integrating computer is practically zero. The data reaches the machine in a continuous flow, one code after the other, without requiring an access from the memory. This permits using an extremely high-speed memory such as coaxial cable delay lines wherein a pulse repetition rate of a few tens of megacycles may be attained. The addition speed in a DDA can thus be increased to 1 million operations per second.

d) Solution of a relatively complex problem in a DDA does not require an internal memory of large capacity. Moreover, there is no need to store the operation codes and memory addresses in the internal memory of a DDA.

e) Depending on the required accuracy, the digital integrating computer possesses flexibility in computing speed. A given problem may first be solved with a low accuracy but at a high speed. The same problem is then solved with a higher accuracy but longer solution time.

Since the solution time in a DDA is linearly dependent on the required accuracy, the speed of solution of a control system not requiring extremely high accuracy may be increased at the expense of accuracy. An important property of the DDA, illustrating its flexibility, is the possibility of automatic searching for those solutions of a problem that satisfy specified boundary conditions. The DDA rapidly "scans" a large family of curves satisfying the stated problem and automatically chooses the curve that meets the given boundary conditions or that agrees with the given coefficients in the equations to be solved. Such a problem requires much more time on a general-purpose digital computer.

Of considerable promise in increasing the efficiency and operating speed of a large general-purpose arithmetic machine is the use of a serial or parallel DDA in conjunction with a general-purpose digital computer. The latter is used mainly for a large amount of computations, insertion of initial data into the DDA, accumulation and processing of the resulting data, etc. Direct, real-time control of an object and the computations required for the control purpose are achieved by the DDA.

In solving a problem where a large number of integrators is required, it is more economical to use a serial DDA in which integrations are performed serially by a single adder. A serial DDA possesses considerable flexibility. For example, the operations of conditional transfer and unconditional transfer from one problem solution to another, automatic change in initial conditions and coefficients, simultaneous solution of several problems at different speeds, etc. can be performed. Particularly deserving of attention is the possibility of generating in a DDA various logic combinations by combining different logic elements in accordance with a given program. These logic operations may be interleaved with computing operations. Hence the computer may be used for simultaneous execution of various operations.

As in the case of an arithmetic machine, the serial DDA has a common memory unit which can be a magnetic drum, delay lines, or ferrite cores. However, this unit is much simpler and more compact than that in an arithmetic machine.

Experimental and tabular values can be inserted by using a ferrite-core function generator. It permits insertion of several score tabular functions of one or two independent variables into the computer. Numbers computed in the DAC as well as in the DDA itself may be used as the argument of these functions. The possibility of using a DDA in solving a complex problem which requires

many integrators may be realized by using the DDA for successive solution of segments of the problem. The result of a problem segment from the DDA is written into the output unit and then re-inserted as the initial data for another problem segment.

For example, in solving the system of differential equations

$$y'' = A \cdot y + B \cdot z \cdot x + D \cos\theta; \tag{1}$$

$$\theta'' = C \cdot \varphi \cdot D \cdot \sin^2 \varphi + Ae^{z\theta}; \tag{2}$$

$$\ln \varphi = E \cdot \varphi' + F \cdot y' + z'''; \tag{3}$$

$$W'' = B \cdot z' + G \cdot W' + D \cdot e^{z \cdot w}; \tag{4}$$

the first three equations may occupy the entire capacity of the DDA. By entering the results z''' after solving equations (1) - (3) into the DDA, equation (4) may then be solved, using z as the input quantity.

The capacity of a digital integrating computer can be expanded several times by increasing the capacity of the storage and providing parallel computing channels. For example, consider a computer which is expanded to have two parallel computing channels with a capacity of 50 integrators each (or a total capacity of 100 integrators). The control unit, designed for 50 integrators, is common to both channels. The problem solution time and the control unit of the computer, despite the two-fold increase in capacity, remain unchanged. It is also possible without any particular difficulty to construct a DDA having several hundred integrators operating with a word length of 20 or more bits. For solving particularly complex problems, several computers are joined in parallel. The newest DDA is an ultra-high-speed parallel computer designed for real-time control and simulation. In view of its recency, possible applications of this computer have not yet been fully investigated.

One advantage of the parallel DDA in comparison with the serial DDA lies in the fact that the problem solution time is determined by the number of iterations. The number of iterations depends on the required accuracy, which for most problems does not exceed 6 to 7 decimal digits, hence in one hour approximately 360 million iterations may be performed in a parallel DDA. At a clock frequency of 10 Mc, this figure can be increased to one billion iterations, or can correspond to an accuracy of 9 decimal places.

The parallel DDA consists of individual standard integrators which are interconnected by a plugboard. As a rule, it does not need a common internal memory or a common control unit; this simplifies its construction and operation. Setting up of the DDA reduces to connecting the individual integrators and is thus more rapid and simpler than the programming in an arithmetic machine. Possible applications of a parallel DDA are considerably increased by the use of multiplier integrators and servo integrators as well as function generators for insertion of tabular and experimental

values. In solving a complex problem, these units decrease the required number of integrators by a factor of 3-4 in comparison with the case where these units are not available; thus, these units provide a considerable economy in electronic equipment.

The individual elements of a parallel DDA (integrators, function generators, multiplier integrators, and servo integrators) may find extremely wide use in automation of many production processes instead of analog integrating and computing devices. For example, digital function generators and integrators may be used in machine-tool control. Individual digital integrators are conveniently used as integrating units, interpolating units, digital filters, multiplier-dividers, and the like.

Since these digital elements possess a degree of accuracy not attainable in analog elements, they are used as elements in automation and remote control systems.

It follows from the above remarks that electronic digital integrating computers, for many practical applications, have substantial technical and economic advantages. Simplicity, compactness, high operating speed and computing accuracy, as well as the possibility of executing a variety of logic operations permit the use of the DDA for control applications, primarily for automation of many production processes. These computers may also be used for scientific-engineering computations and simulation in research and development.

Since electronic computers are gradually becoming instruments for daily use in computation, the DDA permits a simple solution of a class of problems which are of fundamental importance for many fields in science and engineering.

Chapter I

Principles of Digital Integration

1. TYPES OF DIGITAL INTEGRATING COMPUTERS (DDA)

The digital integrating computer or digital differential analyzer (DDA), an analog-digital type of system, may be referred to as a digital analog computer. It successfully combines the principle of solving mathematical problems as it is employed in analog computers and the digital technique of carrying out mathematical operations as it is used in digital computers. As distinguished from "arithmetic" digital computers, all mathematical operations in the integrating digital computer are reduced to a single operation of integration which can be achieved as an operation of addition or subtraction of numbers. These operations are executed in the computer by means of a digital integrator, which replaces the electromechanical or electronic integrator in analog computers. The digital integrating computer is essentially a general-purpose computer and, to a greater or lesser degree, is suitable for solution of various mathematical problems. The DDA is effectively used for solving a system of linear or nonlinear differential equations, algebraic equations, integral equations, trigonometric equations or other problems associated with calculation of continuous functions.

1.1 Serial and Parallel Operations

Digital integrating computers may be divided into two types: serial and parallel.

In the serial DDA the common units are: the memory, control unit, integrating unit and input-output units. The common memory permits the continuous solution of a mathematical problem to be achieved serially, that is, one integration after another by means of only one integrator. Such a serial method of solving a problem by means of a single integrator in an analog computer is, in principle, possible if a suitable analog memory is available.

In a serial DDA the interconnection of integrators into a sequence required for a problem solution is achieved by means of a program which may be fixed, or altered according to the result obtained during the problem-solving process. In other words, the DDA may

achieve conditional or unconditional transfer from one program to another in a manner similar to that achieved in a conventional digital arithmetic computer. From the standpoint of economy, serial DDA's are conveniently used for solving complex nonlinear problems requiring a relatively large number of integrators. They possess wide possibilities for execution of various logic operations (e.g., generation of a sequence of logic operations of AND, OR, NOT; automatic transfer of programs, etc.). The speed of problem solution in a serial DDA is 100-1,000 or more iterations per second, depending on the complexity of the problem.

A parallel DDA consists of a number of individual digital integrators; each integrator is a self-contained unit operating independently of the other integrators. In addition to integrators, the parallel machine uses other, special-purpose units: multiplier integrator, servo unit (as comparator) and function unit.

In a parallel machine, it is not necessary to have a common control unit, a common memory unit and an arithmetic unit. Interconnection of individual integrators is fixed by means of a plugboard. For this reason, the parallel DDA cannot achieve a flexible program of computation and execution of logic operations (for example, automatic transfer of programs during the course of problem solution). Hence the logic capability is extremely limited. The advantage of the parallel DDA is high computing speed of 100,000 iterations per second or more. Of extreme importance is the fact that the speed of problem solution does not depend on complexity of the problem. This speed advantage is achieved at the expense of more integrators and, consequently, leads to a considerable increase in the electronic equipment. Thus, the use of parallel DDA is advisable only in those cases where an extremely high computation rate is necessary.

Combination of the problem-solving principle used in analog computers and the digital technique permits achieving in both types of DDA an essentially new type of digital computer possessing advantages of both analog and digital machines. On the one hand, it is possible to obtain in a digital integrating computer continuity of operation, high operating speed, setup simplicity, convenience in insertion and generation of various functions, and ease of operation of the computer. On the other hand, the use of digital integrators permits a substantial increase in accuracy of computation, in flexibility and general applicability of the computer, and in simplicity of design and economy. To solve a problem in a digital integrating computer does not require a complex programming as required in a conventional digital computer. The DDA may be rapidly changed over to solution of various problems and possesses considerable advantages in flexibility of setup and operation.

The digital integrating computer CRC-105, produced in the United States, operates for six-decimal-digit numbers and has a relatively high computation speed (3600 integrations per second). There are 60 integrators; each integrator contains 6 decimal digits, while computation is performed for 12 decimal digits. The memory is a magnetic drum with 12 tracks for write-in and read-out of number

codes as well as the program for connecting the integrators. Data is inserted into the computer through 12 input channels by means of punched tape from which various experimental functions may be coded. The result of the computations is punched on tape for the 12 channels. All the information is inserted and printed out in decimal notation, and the computation within the machine is executed in binary-coded decimal numbers. There are approximately 400 tubes in the computer.

Since 1950, a number of different types of digital integrating computers containing 20, 60, 100 and more integrators have been developed in the United States for use in research, engineering and various other computations. They are used for solving a system of differential equations describing flight trajectories as well as equations associated with the design, development and control of aircraft, rockets, guided missiles, submarines, etc. Effective use of digital integrating computers in controlling real-time objects is the result of high speed and continuity of computation in these machines and of simplicity in inserting the external information.

The input parameters, given in the form of voltages of continuously varying amplitude and mechanical displacements, are converted into pulses which are applied in sequence at its input. Each pulse corresponds to a definite increment of the physical quantity, while the quantity itself at each instant may be represented by the contents of a counter. Thus, the coded number with which the machine operates is the sum of increments; these increments are represented by pulses in a definite scale. Direct insertion of the coded numbers into the computer as an integrand is also possible.

One of the great advantages of the integrating computer is its ability, in accordance with assigned increments, to achieve continuous solution of a problem with a delay of not more than 0.001-0.01 second. In fact, since the computer for each problem solution operates the increments of the quantities continuously arriving at the inputs, it proves to be possible to attain a computation speed of up to 1,000 problem solutions per second or to obtain 1,000 points of a sought function per second.

Insertion of input data from various sources and interface of the computer with the controlled object may be achieved quite simply with great flexibility in connections and with possibility of changeover by switching. By its nature the integrating computer is an automatic machine in which, due to the wide use of feedback, computation may be checked automatically.

1.2 Basic Computer Operations

In a serial machine the sequence of operations is determined by a coded program. In this program individual time intervals may be used both for computations and for execution of logic operations. The basic operations performed by the computer include (Fig. 1):

(a) Counting by a reversible counter of external or internal pulses. The counter may have up to 7 inputs and a capacity determined by the program (Fig. 1a).

(b) Summing of coded numbers (i.e., addition and subtraction) as well as comparison of two numbers (Fig. 1b).

(c) Integration, used for various computations, algebraic and trigonometric operations, function generations, etc. (Fig. 1c).

(d) Comparison of quantities (servo integrator operation) in summing several increments and then comparing the summed increments (Fig. 1d). The servo integrator (Fig. 1e) is used as a digital servomechanism with feedback loops.

(e) Logic operations consisting of the basic AND, OR and NOT logic functions and combinations thereof in a programmed sequence. These operations also include automatic switching of the problems to be solved, varying the solution rate of different problems, changing coefficients, changing the sign of functions, limiting variable quantities, etc. (Fig. 1f).

(f) Decoding of the code inserted into the machine for control purposes. The decoding is a special logic operation which can be performed by the computer (Fig. 1g).

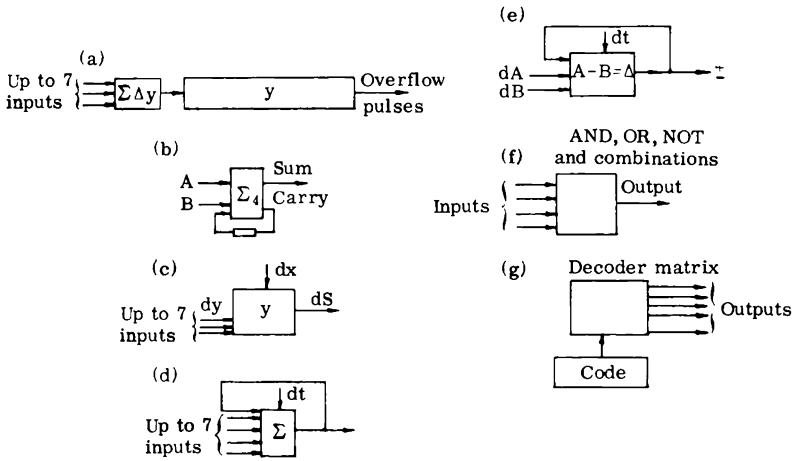


FIG. 1. BASIC COMPUTER ELEMENTS:
(a) reversible pulse counter with several inputs; (b) serial adder; (c) integrator; (d) pulse-increment adder; (e) servo integrator (comparator); (f) logic circuits; (g) decoder.

1.3 Advantages and Disadvantages of Serial Computer

Below are listed the principal advantages of digital integrating computers:

(a) Reliability due to fewer electronic components;

(b) Automatic nature of the machine operation and use of feedback;

(c) Flexibility of the control unit and simplicity of switching the computer to solving various problems;

(e) High computing speed;

(f) Accuracy of computation (higher than that of electromechanical machines);

(g) Absence of accumulation of error in integration;

(h) Continuous delivery of computed data;

(i) Simplicity and flexibility in obtaining input data from various information sources;

(j) Convenience in representation and conversion of input data obtained from graphs, tables, instrument readings, etc.;

(k) Ease of operation (computer maintenance and personnel training);

(l) Low power requirements.

The disadvantages of digital integrating computers are:

(a) Linear dependence of accuracy on solution time; the result is that the accuracy, for all practical purposes, is limited to 6 or 7 decimal digits;

(b) Initial computer processing of the input data in control applications requires a certain amount of time, depending upon the computing speed and upon the amount of difference between the input data and the initial conditions in the computer;

(c) Insufficient speed in processing functions whose magnitudes are "jumped" and excessive build-up time for those applications requiring a control computer with high dynamic requirement. This disadvantage is due to the fact that the computer operates with a fixed increment which is equal to the unit increment and that the generation of any sizable increment requires a large number of steps.

1.4 Incremental Computer

Recently digital integrating computers operated with variable increments have appeared; in such computers the disadvantage of the DDA—slow build-up time (due to magnitude-jump functions)—is eliminated. In these so-called "incremental" computers with variable increment, the increment may take different values approximated by one of the numbers of the series $\pm 2^0$, $\pm 2^1$, $\pm 2^2$, $\pm 2^3$, $\pm 2^4$, etc., wherein the error from approximation of the increment in powers of two is taken into account in the next step of computation and does not affect the accuracy.

Digital integrating computers with variable increments perform many mathematical operations such as addition and subtraction, multiplication, division, integration, differentiation, squaring, extracting roots, and other combined operations; it solves various mathematical problems according to a given program.

In the iteration method of solution, wherein summation of increments (integration) is performed, this computer is similar to the digital differential analyzer but with a variable increment. The solution time in an incremental computer is considerably shortened due to:

(a) Execution of iterations with variable increment;

(b) Simultaneous execution of several arithmetic operations by means of several adders and multipliers;

(c) Execution of multiplication and division in one addition cycle; herein the memory access time is practically zero;

(d) The number of iterations required for solution of a problem depends on the magnitude of the jump in the function. For continuous functions, the solution of a problem requires one iteration;

(e) Generation of the initial conditions is achieved tens of times more rapidly than in a fixed-increment DDA;

(f) Such complex operations as computation of sin x, arc sin x, $x \cdot y$, e^x, $\ln x$, $\frac{x}{y}$, $\sqrt{x}$, etc., are performed in the computer within approximately 1 to 2 addition cycles, whereas such complex operations in a conventional computer require a relatively long series of simple operations. Execution of such complex operations in the DDA requires several integrators.

Thus, the advantages of digital incremental computers with variable increment are:

(a) Extremely high repetition rate of computing operations;

(b) Excellent ability to perform repetitive computing cycles;

(c) Acceptable build-up time in performing magnitude-jump functions;

(d) Satisfactory reliability;

(e) Medium size, weight and number of components;

(f) Adequate capability in logic operations.

Table 1 below lists comparative data for a digital differentiating analyzer (DDA), a variable-increment digital integrating computer (VIDIC) and a digital arithmetic computer (DAC).

TABLE 1

Comparative Characteristics for DDA, VIDIC, and DAC
(Published data (Ref. [19]))

	DDA	VIDIC	DAC
Bandwidth, cps	25	25	2.5
Setup time, sec	300	10	0.05
Time for computing one result	300	10	0.4
Accuracy, %	0.1	0.1	0.1
Size (DAC as unity)	1/2	3/4	1
Weight (DAC as unity)	—	3/4	1
No. of components (DAC as unity)	1/4	1.5	1
Downtime, hr	0.5	—	—
Average stoppage per 1000 hrs:			
(a) Total	10	40	50
(b) Serious	5	7	2
Average failures per 1000 hrs	1	4	5
No. of malfunctions per 1000 hrs	6	11	7

2. PRINCIPLES OF INTEGRATION

2.1 Approximate Representation of an Integral

Digital integrating computers employ an approximate representation of the integral of an function as the sum of the areas of elementary rectangles, each of which corresponds to a definite increment Δx of the independent variable x.

Let us assume (Fig. 2) that we are required to find the integral of a given function $y = f(x)$. This integral represents the area bounded by the curve $y = f(x)$ and the abscissa, but the integral may be approximated as the sum of the areas of the elementary rectangles. The height of each of the rectangles is the current ordinate y and the base is the increment Δx of the independent variable x; each increment is obtained by dividing the entire range of change in variable x into equal increments.

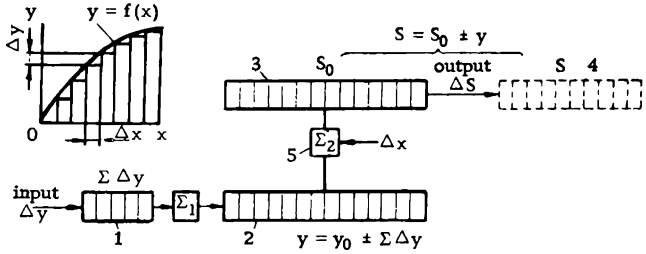


Fig. 2. Block diagram of digital integrator.

Then the integral S is

$$S = \int_{x_0}^{x} y dx = \lim \sum_{i=1}^{i=n} y_i \cdot \Delta x_i + S_0 \approx \sum_{i=1}^{i=n} y_i \Delta x_i + S_0, \tag{1}$$

where

$$y = f(x),$$

and

$$\Delta x_i = x_{i+1} - x_i$$

is the increment of independent variable x.

We shall assume that Δx is equal to 1, (i.e., each of the increments is taken as unity). Then

$$S \approx \sum_{i=1}^{i=m} y_i + S_0 \tag{2}$$

where S_0 is the initial value of the integral.

Thus, to compute the integral it suffices to add all the ordinates corresponding to each of the elementary rectangles. Consequently, the integration process reduces to summation of the numbers representing the ordinates.

Each ordinate is equal to the preceding ordinate plus (or minus) the ordinate increment Δy_i for the interval $\Delta x_i = x_{i+1} - x_i$. Hence the value of ordinate y_i may be computed by adding to each preceding ordinate the increment Δy. For example,

$$\left.\begin{aligned} y_1 &= y_0 + \Delta y_1, \\ y_2 &= y_1 + \Delta y_2, \\ &\cdots\cdots\cdots \\ y_n = y_{n-1} + \Delta y_n &= y_0 + \Delta y_1 + \Delta y_2 + \ldots + \Delta y_n. \end{aligned}\right\} \qquad (3)$$

Hence it is seen that the current value of the ordinate y_i may be obtained by accumulating all increments of the ordinate up to the increment Δy_i.

2.2 Wheel-and-disk Integrator

An integrator for computing ordinate y and the value of integral S is represented by a block symbol (Fig. 3a) having two inputs and one output. Integrand change dy is applied to one of the integrator inputs, and independent variable change dx to the other input. The integrator gives the change dS as the output.

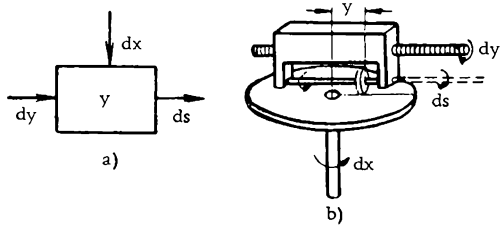


Fig. 3. Wheel-and-disk integrator.

If we compare the digital integrator (Fig. 2) with the wheel-and-disk integrator (Fig. 3b), the increment of the integrand Δy (Fig. 2) corresponds to a unit angular displacement of the screw (Fig. 3b) which displaces the integrating-wheel carriage. It is evident that displacement y of the wheel from the center of the disk corresponds to the current ordinate of the curve $y = f(x)$. The angular displacement of the disk shaft represents the independent variable x. Increment Δx corresponds to the unit angular rotation of the disk shaft which is in frictional contact with the integrating wheel. The unit angular displacement of the integrating wheel corresponds to the integral increment ΔS, since the total angular displacement of the wheel shaft represents the integral

$$S = \int y dx.$$

All three quantities x, y and S are, respectively, the accumulations of dx, dy, and dS which are either the input or the output of the integrator.

2.3 Operation of the Digital Integrator

The operation of the digital integrator (Fig. 2) is now described. Assume that increments Δx, Δy and ΔS are given in the form of individual pulses. The reversible counter 1 counts the incoming pulses. For each integration step, the counter accepts pulses from several input channels and the number of pulses accumulated by the counter is regarded as an increment of the integrand, where the increment is the sum of several elementary increments, i.e., $\sum \Delta y$. This increment would correspond to a change of angular displacement of the screw of the wheel-and-disk integrator if this change resulted from simultaneous angular increments from several input shafts through differential gears.

For each integration step, the number $\sum \Delta y$ stored in counter 1 is summed with the number y_0 stored in register 2 by means of the $\sum_1$ adder. As a result of addition (or subtraction) a new ordinate $y = y_0 \pm \sum \Delta y$ is obtained for each integration step.

Also for each integration step, the number y stored in register 2 is added to register 3, wherein is stored the number corresponding to the sum of the ordinates (i.e., the value of the integral). Summation of numbers y and S_0 is achieved by the $\sum_2$ adder.

$$S = S_0 \pm y, \tag{4}$$

where S_0 is the number of the integral initially stored in register 3. Summation is performed each time the input Δx of the integrator receives a pulse representing an increment of the independent variable x. Both registers 2 and 3 have the same number of n bits.

During summation, register 3 may overflow (i.e., a carry pulse from the most significant bit). The overflow pulse of register 3 is considered to be the increment ΔS of integral S at the integrator output. The integral increment pulses ΔS are accumulated in counter 4, shown in the figure by dotted lines. Register 3 and counter 4 may be regarded as two parts of a single register having $2n$ bits. Register 3 holds the less-significant bits of integral S and counter 4 holds the more significant bits located at $(n+1)$ to $2n$ bit positions. In such an arrangement, integral S is limited to have twice as many bits as the integrand register 2. For example, let x vary from 0 to 1000000_2 with an interval of $\Delta x = 1$ and let integrand $y = y_0 = 1000000_2$. With each summation of y and S, an overflow pulse ΔS occurs. The number of all overflow pulses stored in register 4 is 1000000_2. The value of integral S is 1000000^2.

The process of accumulating the overflow pulses ΔS from the integrator output by register 4 is also an integration process. In general, the integral increment ΔS is

$$\Delta S = K \cdot y \cdot \Delta x \tag{5}$$

where K is a constant scale factor. For binary numbers

$$K = \frac{1}{2^n} \tag{6}$$

and n is the number of bits of register y or S, wherein S_0 denotes the number in register 3 (the remainder of the integral).

Coefficient K is the scale factor of the digital integrator. It signifies that, for $y = 1$ and $\Delta x = 1$, 2^n summations (or 2^n steps) are required in order to obtain one overflow pulse ΔS at the output of register 3. If y is equal to 2^n and $\Delta x = 1$, then there will be an overflow pulse for each of the integration steps. In using decimal numbers the coefficient K is

$$K = \frac{1}{10^n}, \tag{7}$$

where n is the number of decimal digits of the register.

Converting from increments Δx, Δy and ΔS to derivatives of x, y and S in time, formula (5) may be written as follows:

$$\frac{dS}{dt} = K \cdot y \frac{dx}{dt} = K \cdot \left(\int \frac{dy}{dt} dt \right) \frac{dx}{dt} \tag{8}$$

or in the form of an integral

$$S = K \int y dx \ . \tag{9}$$

2.4 Errors of the Digital Integrator

The value of integral S obtained in digital integrating computers is approximate due to errors arising from the following reasons:

(a) The approximation of the integral by summing the rectangular areas contributes error. The larger the number of rectangles (and of integration steps), the more accurately the sum of the rectangular areas approaches the value of the integral in formula (9). If the quantity Δx and the number n are properly chosen, one can obtain an error less than a given value ε, that is,

$$\int_{x_0}^{x} y dx - \sum_{i=1}^{i=n} y_i (x_{i+1} - x_i) < \varepsilon. \tag{10}$$

(b) The limited capacity of registers y and S leads to round-off of the result. In order to increase the accuracy in a digital

integrating computer, calculation of the sum of the ordinates is performed with a double number of digits, that is, the value of integral S is fixed by two registers 3 and 4 having a total number of 2 n digits.

(c) The method of coding overflow pulses in a computer using binary numbers, as will be shown below, can cause an error in computing the integral by one least significant bit of register 2 or 4. This error must not be accumulated in the computing process. It may be eliminated by using the so-called ternary method of coding the overflow and increment pulses, wherein a positive increment corresponds to +1, a negative increment to −1, and the absence of increment to 0. However, coding of the three possible values for the increments complicates the computer implementation.

The error in rectangular integration may also be decreased by using the trapezoidal rule (Fig. 4a) wherein the curve $y=f(x)$ at each interval Δx is approximated by a chord. This rule is equivalent to summation of the areas of "mean" rectangles as shown in Fig. 4b; each rectangle has a mean ordinate approximated by

$$y_0 + \frac{1}{2}\Delta y.$$

Then the increment of the integral (the area of an elementary rectangle) is

$$\Delta S = \left(y + \frac{1}{2}\Delta y\right) \cdot \Delta x. \tag{11}$$

As is known, the trapezoidal method reduces the error of integration by a factor of about 10 in comparison with the Euler method of rectangles.

However, use of the trapezoidal method is significant only where other errors are sufficiently small. In coding increments Δx, Δy, ΔS in a binary number system, wherein an increment may be positive (+1) or negative (−1) and zero corresponds to the sum of +1 and −1, the integration error may amount to a least significant bit in the y or S register; this error is considerably larger than the error arising from approximating the sum of

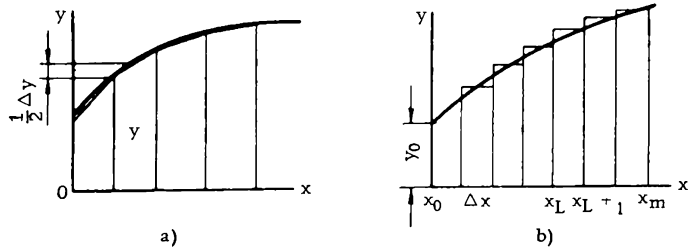


Fig. 4. Trapezoidal integration.

rectangles to the integral. Hence the use of the trapezoidal method is justified if one also uses the ternary method of representing increments $\Delta x, \Delta y$, and ΔS.

3. TRAPEZOIDAL INTEGRATION

3.1 Integration by the "Mean" Rectangle Method

In integration by using the trapezoidal rule, the integral, as is known, is approximated by

$$\int_{x=x_0}^{x_n} F(x)\,dx \approx \frac{x_n - x_0}{n}\left(\frac{y_0 + y_n}{2} + y_1 + y_2 + \ldots + y_{n-1}\right). \qquad (12)$$

Similar accuracy in integration may be obtained by summing the areas of rectangles having at each interval a mean ordinate (Fig. 5) of

$$y_m = y + \frac{1}{2}\Delta y, \qquad (13)$$

where y is the value of the current ordinate and Δy is the ordinate increment for the given interval.

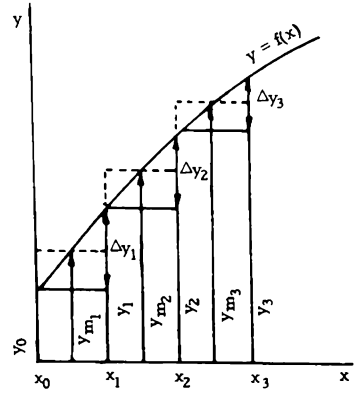


Fig. 5. Integration by the "mean" rectangle method.

The approximate value of the interval will then be

$$\int_{x=x_0}^{x_n} y\,dx \approx \frac{x_n - x_0}{n}\left(y_{m1} + y_{m2} + \ldots + y_{mn}\right). \qquad (14)$$

The maximum error Δ of this integration is

$$\Delta = \frac{(x_n - x_0)^3}{24n^2} |f''(x)|_{\max}, \tag{15}$$

where $|f''(x)_{\max}|$ is the maximum value of $|(f''(x)|$ in the interval from x_0 to x_n.

Integration by using formula (14), which is close in accuracy to the trapezoidal formula, consists of three summing operations:

(a) The next ordinate value y^* is determined by adding the increment Δy at the integrator output, i.e.,

$$\begin{aligned} y_0 \pm \Delta y_1 &= y_1^*, \\ y_1 \pm \Delta y_2 &= y_2^*, \text{ etc.} \end{aligned} \tag{16}$$

(b) Half of the ordinate increment is added to the initial value of the ordinate, i.e.,

$$\left.\begin{aligned} y_0 \pm \frac{\Delta y_1}{2} &= y_{m1}^{\cdot}, \\ y_1 \pm \frac{\Delta y_2}{2} &= y_{m2}^{\cdot}, \\ y_2 \pm \frac{\Delta y_3}{2} &= y_{m3}^{\cdot}, \end{aligned}\right\} \tag{17}$$

etc.

(c) The resulting mean ordinate values are summed in the accumulator (S register). The sum of the mean ordinate values is taken as the approximate value of the integral,

$$S \approx y_{m1}^{\cdot} + y_{m2}^{\cdot} + y_{m3}^{\cdot} + \ldots + y_{mn}^{\cdot}. \tag{18}$$

3.2 Block Diagram

The block diagram for integration by the mean-ordinate summation method is given in Fig. 6. Increments Δy_a, Δy_b, Δy_c arriving from different channels at input Δy of the integrator are algebraically summed by means of the adder $\sum \Delta y$, giving the output $\sum \Delta y$ in direct or complement form. By means of a single-digit adder $\sum_1$, this output is summed with the contents of the y register, giving the new value of y^* which is again stored in the y register.

By means of the adder $\sum \Delta y/2$, the sum $\sum \Delta y$ is divided by two, and the result is summed with the number y (the calculated initial ordinate of a given interval) by the adder $\sum_2$. Thereby, we obtain the mean value of ordinate $y_m^{\cdot}$ for the given interval

$$y_m^{\cdot} = y \pm \frac{\Sigma \Delta y}{2}. \tag{19}$$

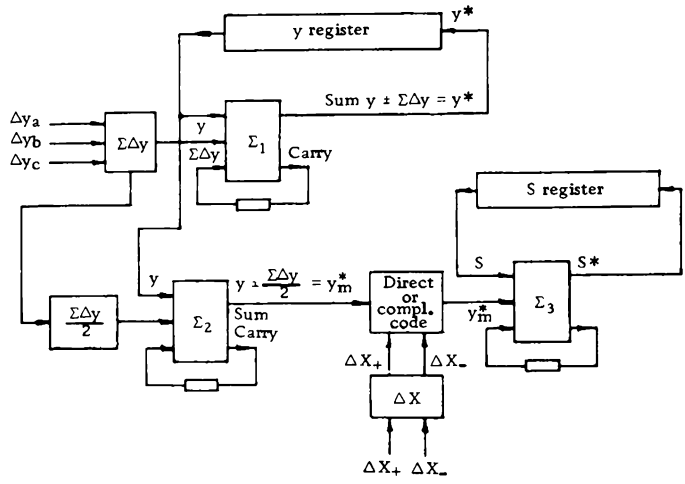


Fig. 6. Integration by the mean rectangle method.

Depending on the code of the increment of the independent variable Δx ($+1$ or -1) stored in the Δx block, the number y_m^* arrives at single-digit adder $\sum_3$ in direct or complement form. Adder $\sum_3$ sums ordinate y_m^* with the contents of the S register, where the remainder of integral S from the previous summation is held. Thus,

$$S^* = S \pm y_m^*. \tag{20}$$

The increment ΔS at the integrator output ($+1$ or -1) is delivered by means of a special circuit using the ternary system of increment coding (see Section 6). The capacity of registers y and S in using the mean-ordinate method of integration is one bit longer than that of registers y and S in integration by using the Euler formula. This is necessary in order to accommodate the increment $\frac{\Delta y}{2}$. For this purpose the summation of increment Δy in adder $\sum_2$ begins with the least significant digit to that of code number of y. This increment Δy in adder $\sum_2$ begins with the second least signigicant digit to the least significant digit of number y. Thus, increments of $\frac{\Delta y}{2}$ and Δy are obtained.

4. BINARY CODING

In digital integrating computers, the operations are performed with fixed-point fractional numbers having a range from $+1$ to -1 All quantities which are entered into the computer and with which the operations are performed are expressed, with a scale, in the

form of numbers ranging from +1 to −1. Depending on the capacity of the integrator registers, the numbers are represented by a fixed number of binary or decimal digits. For example, if a register holds a six-digit decimal number, the fractional part of the number contains six decimal digits.

4.1 Binary System

The binary system is most convenient for coding the numbers in a computer. In this system (Fig. 7) the largest positive number 0.99999 corresponds to the number [1]111...11 in the binary system. The enclosed digit is the sign digit. The sign digit of positive numbers is 1 and that of negative numbers is 0. Binary code [0]000...0 represents the largest negative number −1. Table 2 lists the codes for several numbers in the adopted binary system.

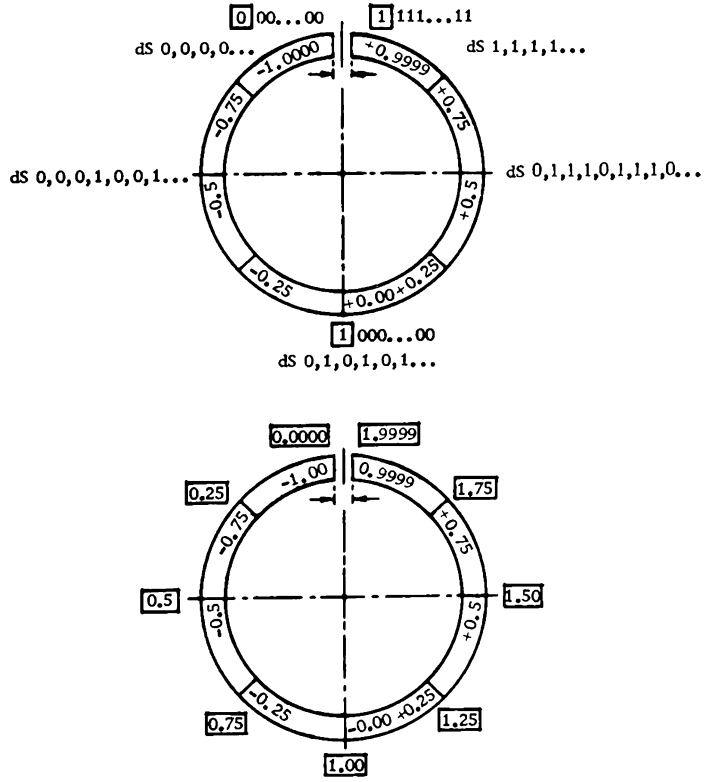


Fig. 7. Number coding system.

TABLE 2

Coding of Numbers in y and S Registers

Decimal numbers	Binary code
+ 8	[1] 000 . . . 001000
+ 7	[1] 000 . . . 000111
+ 6	[1] 000 . . . 000110
+ 5	[1] 000 . . . 000101
+ 4	[1] 000 . . . 000100
+ 3	[1] 000 . . . 000011
+ 2	[1] 000 . . . 000010
+ 1	[1] 000 . . . 000001
+ 0	[1] 000 . . . 000000
−8	[0] 111 . . . 11000
−7	[0] 111 . . . 11001
−6	[0] 111 . . . 11010
−5	[0] 111 . . . 11011
−4	[0] 111 . . . 11100
−3	[0] 111 . . . 11101
−2	[0] 111 . . . 11110
−1	[0] 111 . . . 11111
−0	[0] 111 . . . 00000

Note that the addition of a one pulse to the least significant digit of the largest positive number [1] 11111...1 converts the number to [0] 000...00, i.e., the largest negative number. This conversion produces an overflow in the y register.

The above described binary system is used only for y and S registers. In the counter and $\sum\Delta y$ register, summation of the

Δy increment pulses is represented by ordinary binary numbers where the sign for a positive and a negative number is given respectively by the code 0 and 1. These numbers are held in the counter and the $\sum\Delta y$ register (Table 3).

TABLE 3

Coding of Increments in $\sum\Delta y$ Counter

Decimal number	Binary code (for 3 bits)	Decimal number	Binary code
+4	0 100	−1	1 111
+3	0 011	−2	1 110
+2	0 010	−3	1 101
+1	0 001	−4	1 100
+0	0 000	−5	1 011

A correct result is obtained in adding or subtracting the numbers in $\sum\Delta y$ and y which have different binary codes. For example:

In the decimal system:

$$\begin{array}{rr} & y = +4 \\ + & \sum\Delta y = -2 \\ \hline & y + \sum\Delta y = +2 \end{array}$$

In the binary system:

$$\begin{array}{rr} & \boxed{1}000\ldots00100 \\ + & \boxed{1}111\ldots11110 \\ \hline & \boxed{1}000\ldots00010 \end{array}$$

The above described binary system has one important property which will be used later in the servo integrator (see Sec. 10). This is the addition of +1 (see code in Table 3) to the largest possible number y $\boxed{1}$ 111...1, changing the latter number into the largest negative number, i.e.,

$$\begin{array}{rr} & \boxed{1}111\ldots111 \\ + & \boxed{0}000\ldots001 \\ \hline & \boxed{0}000\ldots000 \end{array}$$

On the other hand, addition of −1 (see code in Table 3) to the largest negative number $y = \boxed{0}\,000\ldots00$ changes the number into the largest positive number, i.e.,

$$y\ \boxed{0}000\ldots000$$
$$\sum\Delta y = -1\ \boxed{1}111\ldots111$$
$$\boxed{1}111\ldots111$$

4.2 Decimal System

In integrating computers designed for engineering calculations and processing of experimental data, all information is entered into the computer in a convenient decimal system. The results of computation are also obtained in decimal form. In order to avoid the use of complex binary-to-decimal converters, all computations are also performed in the decimal system (more accurately, binary-coded decimal numbers).

In this decimal system, numbers ranging from+0.999...99 to −1.0000...00 are represented in the machine by 1.9999...99 to 0.000...00, respectively. Thus, positive numbers in the machine have sign digit 1 (left of the decimal point) and negative numbers, sign digit 0. Positive numbers range from 1.9999 to 1.0000 and negative numbers from 0.0000 to 0.9999.

Note that the coded numbers 0.00...00 to 1.99...99 are the decimal complement of 1.000..00 (i.e. addition of 1 to the positive or negative decimal number). For example, the number +0.6321 is coded in the machine as 1.6321. The negative number −0.6321 is coded as 0.3679.

For a four-digit decimal number, negative number −.0001, is represented in the machine by 0.9999 and the positive number +.0001 by 1.0001. Summation of the numbers y and $\sum\Delta y$ in the y register is achieved either in the decimal or binary system. The number code for $\sum\Delta y$ ranges from 0 to 9 in the least significant digit of a decimal counter accumulating Δy increment pulses. The sign digit code in the $\sum\Delta y$ counter is the one's complement of that in the y register. Thus,+1 in the least significant digit position of the $\sum\Delta y$ counter is written as $\boxed{0}$.0001; −1, as $\boxed{1}$.9999; −2, as 1.9998; etc.

For example, the following addition,

$$+\quad \begin{array}{r} y = +4 \\ \sum\Delta y = -2 \\ \hline y + \sum\Delta y = +2 \end{array}$$

or in the decimal system of coding,

$$\begin{array}{l} 1.0004 \\ 1.9998 \\ \hline 1.0002, \text{ i.e., the number } +2. \end{array}$$

5. THE INTEGRATION OPERATION

As was explained in Sec. 2, each integration step in a digital integrator is reduced to two summing operations.

(a) The summation of the sum of increments $\sum \Delta y$ with the initial integrand y_0 in obtaining the new value of y:

$$y = y_0 \pm \sum \Delta y. \tag{21}$$

(b) The summation of the ordinates y (accumulation of integral S), or summation of the new value of integrand y with the previously accumulated value of integral S_0, that is,

$$S = S_0 \pm y. \tag{22}$$

5.1 Accumulation of Integral S

The first operation was discussed in detail in Sec. 2. Let us now discuss the second operation with respect to integration.

The numbers S_0 and S in registers 3 and 4 (Fig. 2) are coded in the same manner as the number y in register 2, the only difference being that the number S_0 has one bit to the left of the sign bit, which may be 0 or 1, whereas in the y register this digit is always 0. Choice of the code in this highest digit of the S_0 register depends on which operation, addition or subtraction, must be performed with the overflow (increment) ΔS at the output of the integrator. If the integrator output must not change sign (i.e., increment ΔS must subsequently be added with other increments at the input of the next integrator), a 0 is placed after the sign digit. If it is necessary to change the sign of ΔS, 1 is placed in this position.

For example, the number $S_0 = +13$ is written as:

$$\textcircled{0}\ \boxed{1}000\ldots001101,$$

where the circle shows the overflow digit of the register.

Upon sign reversal of ΔS to $-\Delta S$, code 1 is entered in the overflow digit of the S_0 register, i.e.,

$$\textcircled{1}\ \boxed{1}000\ldots001101.$$

It is easily seen that in this case the overflow pulse code is complemented. For example, upon adding $S_0 = +13$ to $y = +7$ in the first case we obtain

$$\begin{array}{rl} S_0 & \textcircled{0}\ \boxed{1}000\ldots001101 \\ y & \textcircled{0}\ \boxed{1}000\ldots000111 \\ \hline \text{Overflow Pulse Code 1} & \textcircled{1}\ \boxed{0}000\ldots010100 \end{array}$$

In the second case

S_0	① [1] 00...001101
y	⓪ [1] 00...000111
Overflow Pulse Code 0	⓪ [0] 00...010100

5.2 Overflow Pulse Sequences

In integrating computers the increments ΔS, Δy and Δx are usually represented in the binary system, since it is the simplest to build a memory with two states 1 and 0. Code 1 corresponds to the presence of an overflow (or increment) pulse, and denotes a positive increment of +1. Code 0 corresponds to the absence of an overflow pulse and denotes an increment of —1. The absence of an increment or a zero increment at the integrator output corresponds to the code sequence 010101...01. This is decoded as the sum —1+1—1+1..., that is, 0.

Several examples are now given to explain the addition and subtraction of numbers S_0 and y for each integration step and to show sequences of overflow pulses corresponding to different integral increments.

Example 1. Let the initial value of the integral be $S_0 = 0.0$ (or code [1] 000...00) and the integrand y be +1 (or code [1] 000...001). Integrand y will remain constant during the integration. Then for each step of integration, the addition of y to the S_0 register is to add one.

In the first step of integration,

S_0	[1] 0000...00
y_0	[1] 000 ...01
Overflow Pulse Code 1	[0] 000 ...01

In the second step,

S_0	[0] 00...01
y_0	[1] 00...01
Overflow Pulse Code 0	[1] 00...10 etc.

Thus, a sequence of overflow pulse codes 1, 0, 1, 0,... appears at the integrator output, which corresponds to the integral increment +1-1+1... (i.e., zero).

This sequence of overflow pulses continues until a 1 appears in the highest digit of the S_0 register, which will occur after 2^n additions.

It is thus seen that if we alternately add and subtract the same number y to or from the initial value of integral S_0 (i.e., the case in which the integral changes sign and the integral increment is equal to zero), a sequence of overflow pulses 1, 0, 1, 0, etc. appears at the integrator output.

Example 2. Assume that y is the largest positive number $\boxed{1}$ 111...11 and remains constant while S_0 is equal to zero.

Then, the first step of integration gives

	$\boxed{1}$ 00...00
	$\boxed{1}$ 11...11
Overflow Pulse Code 1	$\boxed{0}$ 11...11

and the second step of integration gives

	$\boxed{0}$ 11...11
	$\boxed{1}$ 11...11
Overflow Pulse Code 1	$\boxed{0}$ 11...10 etc.

Thus, the addition of the largest positive integrand produces a sequence of overflow pulse codes 1,1,1,1, etc.

Similarly, the addition of the largest negative integrand (y = $\boxed{0}$ 000...00) gives a sequence of overflow pulse codes 0,0,0,0, etc., which is decoded in the machine as a sequence of negative increments —1, —1, —1, —1, etc. Since the sequence of overflow pulses at the integrator output characterizes the rate of increase or decrease of the integral, the largest possible integrand y corresponds to a maximum rate of increase in the integral which is characterized by a sequence of overflow codes 1, 1, 1, 1, etc. and the largest negative integrand corresponds to a maximum negative overflow rate characterized by a sequence of codes 0,0,0,0 (i.e., —1,—1, etc.).

When the integrand has zero value (i.e., code $\boxed{1}$ 000...00), there is a zero rate of increments of the integral, i.e., the sequence of overflow pulse codes 1,0,1,0,1,0, etc. We may conclude that the value of the integrand corresponds to the rate of increase of the integral if the integrand remains constant. A variable integrand corresponds to a variable rate of increase of the integral. This is shown by the following two examples.

Example 3. Let the coded decimal integrand (see Fig. 7) be $y = 0.5$, (i.e., one half of the largest negative value). Addition of coded decimal y and S gives:

(a) in the first step of integration

$$\begin{array}{r} S_0 \\ y \end{array} \begin{array}{r} 0.0 \\ +\ 0.5 \end{array}$$

Overflow pulse code 0 0.5 sum

(b) in the second step

$$\begin{array}{r} S_0 \\ y \end{array} \begin{array}{r} 0.5 \\ +\ 0.5 \end{array}$$

Overflow pulse code 0 1.0 sum

TABLE 4

Sequences of Overflow Pulse Codes at Integrator Output for 8 Steps of Integration

Value of function y in decimal code	Sequences of overflow codes	Increment sum after 8 steps of Integration	Binary Code
y = 0	Codes 0 0 0 0 0 0 0 0 Equivalent increments −1−1−1−1−1−1−1−1	−8	[0]00...00 largest negative value
y = 0.5	Codes 0 0 0 1 0 0 0 1 Equivalent increments −1−1−1+1−1−1−1+1	−4	
y = 1.0	Codes 0 1 0 1 0 1 0 1 Equivalent increments −1+1−1+1−1+1−1+1	0	[1]000...00 zero value
y = 1.5	Codes 0 1 1 1 0 1 1 1 Equivalent increments −1+1+1+1−1+1+1+1	+4	
y = 1.9999	Codes 1 1 1 1 1 1 1 1 Equivalent increments +1+1+1+1+1+1+1+1	+8	[1]1111...11 largest positive value

(c) in the third step

$$\begin{array}{rr} S_0 + & 1.0 \\ y + & 0.5 \\ \hline \text{Overflow pulse code 0} & 1.5 \end{array}$$

(d) in the fourth step

$$\begin{array}{rr} S_0 + & 1.5 \\ y + & 0.5 \\ \hline \text{Overflow pulse code 1} & 0.0, \text{ etc.} \end{array}$$

Hence it is seen that with coded decimal $y = 0.5$ the sequence of overflow pulse codes at the integrator output is: 0,0,0,1,0,0,0,1, etc. This sequence corresponds to one half of the largest negative rate of increase of the integral, since there are only two negative increments ($-1\ -1\ -1+1=-1 \cdot 2$) in four integration steps instead of four negative increments (which corresponds to the largest negative rate). This and other similar results are shown in Table 4.

5.3 Coding of $\Delta x, \Delta y$ and ΔS

Due to the fact that the increment pulse codes have only two values, 0 or 1, the accumulated sum of increments of the integral may differ from the correct sum by $+1$ or -1. For example, for an increment ΔS of zero to the S register for each step of integration, the integral output of $+1$ or -1 means for each odd step of integration an error of $+1$ in the least significant digit. This error, oscillating around zero, may be eliminated by use of the ternary system of coding, wherein increments of $+1$, 0 and -1 are coded by three different codes.

Ordinate summation $S_0 \pm y$ occurs each time an increment pulse Δx of independent variable x appears. For each step of integration there is at input Δx only one pulse (one increment of Δx) but there may be at input Δy several pulses from different channels. The increment pulse code of Δx, as that of Δy and ΔS, has two values 1 and 0 ($\Delta x = \pm 1$). The values of 1 and 0 at input Δx correspond respectively to the addition $S_0 + y$ or the subtraction $S_0 - y$. Alternate addition and subtraction (i.e. $S_0 \pm y$) signify a Δx equal to 0. In this case the value of the integral does not vary. The identical coding of increments Δx, Δy and ΔS of the integrator permits their interconnection in any manner. It is possible to feed back the output of an integrator to one of its inputs.

6. TERNARY METHOD OF CODING INCREMEMENTS

The above described method of coding increments Δx, Δy and ΔS with two values 1 and 0 (where 1 and 0 correspond respectively

to +1 and −1) reduces the possible accuracy of computation since this method permits an error of unity in the least significant digit of the register. If there are several inputs Δy at the integrator the possible error may have a fluctuation greater than unity in the least significant digits of the register. The ternary method of coding increments avoids the fluctuation error of +1 or −1 and considerably increases the accuracy in the case of trapezoidal integration. In the ternary method of coding, the positive increment corresponds to +1, the negative increment to −1 and the absence of increment to 0. Storage of increments in the ternary system requires separate elements for storing positive and negative increments. Thus, there are two lines for each of the inputs and outputs of the integrator. One line transmits the +1 pulse and the other line the −1 pulse; the absence of pulse on both lines transmits a zero increment.

6.1 First Method

Let us examine the means by which we may obtain at the integrator output increment pulses ΔS corresponding to the ternary system.

The first rule is that if both numbers y and S are positive, overflow of the S register from addition of y and S occurs only when the addition results in a carry from the most significant number digit to the sign digit. Note the sign digits of both positive numbers y and S are 1 (see Table 2).

Example 1.

$S = +8$ code; $\boxed{1}1000$

$y = +7$ code; $\boxed{1}0111$

Increment $\Delta S = 0; +15$ code; $\boxed{2}1111$ (−1)

Here there is no ΔS increment since there is no carry to the sign digit although both numbers are positive.

Example 2.

1 carry

$S = +8; \boxed{1}1000$

$y = +9; \boxed{1}1001$

Increment $\Delta S = +1; +17; \boxed{1}0001$ (+1)

Here we have the increment ΔS since the carry to the sign digit is +1 (i.e., +16).

The second rule is that if both numbers y and S are negative, negative overflow (−1) in the addition of y and S occurs whether there is a carry or not from the most significant number digit to the sign digit. Note that the signs of both negative numbers are 0.

Example 3.

1 111 carries

$S = -15;\boxed{0}0001$

$y = -1;\boxed{0}1111$

Increment $\Delta S = -1$; $-16;\boxed{1}0000$ (0)

The result is correct since there is an overflow —1 (i.e., —16) and the number in the S register is 0 (see Table 2)

Example 4.

$S = -8;\boxed{0}1000$

$y = -9;\boxed{0}0111$

Increment $\Delta S = -1$; $-17;\boxed{0}1111$ (—1)

The third rule is that if positive and negative numbers are added, the presence of a carry from the most significant number digit to the sign digit means that the increment ΔS is +1. Conversely, the absence of this carry denotes an increment of —1.

Example 5.

$S = -8;\boxed{0}1000$

$y = +7;\boxed{1}0111$

Increment $\Delta S = -1$; $-1;\boxed{1}1111$ (+15)

Since the carry is absent, increment ΔS is —1. The result is correct since there is no overflow ($\Delta S = -1$) and the code in S register is +15 (—16 + 15 = —1).

Example 6.

1 carry

$S = +15;\boxed{1}1111$

$y = -8;\boxed{0}1000$

Increment $\Delta S = +1;+7;\boxed{0}0111$ (—9)

Example 7.

$S = +7;\boxed{1}0001$

$y = -12;\boxed{0}0100$

Increment $\Delta S = -1$; $-11;\boxed{1}0101$ (+15)

Since there is no carry to the sign digit, the increment equals —1, and the result is —16 + 5 = —11.

Example 8.

$$S = +10;\boxed{1}1010$$

$$y = -8;\boxed{0}1000$$

$$\text{Increment } \Delta S = +1;+2;\boxed{0}0010\ (-14)$$

With a carry to the sign digit, the increment $\Delta S = +1$. The result of addition is +16—14= +2.

The diagram for obtaining increments in the ternary method is shown in Fig. 8. It is based on the above three rules for obtaining increments of +1, —1, and 0. If the sign digit of even one of the numbers in y and S has code 1, code 1 is stored in flipflop FF_1. During the addition of y and S, the carry from the most significant number digit to the sign digit is stored in flipflop FF_2. If both flipflops FF_1 and FF_2 store code 1, we obtain an increment pulse ΔS of +1 at the output of coincidence gate B_2. We obtain an increment of —1 at the ouput of gate B_3 when the sign of one of the numbers y or S is 1 and there is no carry, or an increment of —1 at the output of gate B_1 when both numbers y and S are negative (code 0). A pulse from the output of FF_1 is an inhibit pulse at gate B_1 but at gates B_2 and B_3 is a pulse for passage of the last pulse P_n. Flipflops FF_1 and FF_2 are cleared by initial pulse H before each addition of numbers y and S.

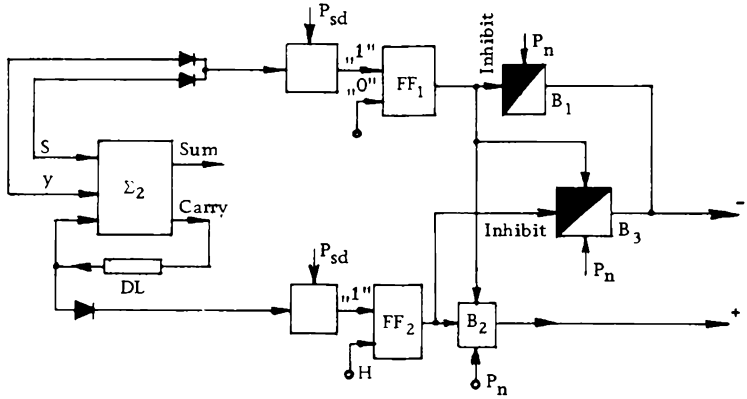


Fig. 8. Generation of increment ΔS by the ternary method (first method).

6.2 Second Method

Another, simpler method of determining the sign of the increment in the ternary method is based on the following rules.

Let positive numbers have the code 0 as sign digit and negative numbers the code 1. We have:

(a) A positive increment of ΔS occurs during addition of numbers y and S when both numbers are positive and when there is a carry of 1 to the sign digit. Otherwise, ΔS is 0.

(b) A negative increment of ΔS occurs during addition when both numbers y and S are negative and when there is no carry to the sign digit. Otherwise, ΔS is 0.

(c) When a positive and a negative number are added, the increment ΔS is equal to zero as that in all other cases except the above cases (a) and (b).

(d) The sign of the sum $S + y$ is determined by the sum itself, except in the above cases (a) and (b) where there is a positive or a negative increment of ΔS. In these two cases, when there is a positive or a negative increment ($\Delta S = +1$ or -1) the sign digit of the register after summation must be respectively the code 0 or 1. In other cases, the sign digit of the sum is correct.

A circuit for generation of the increments and the sign digit is shown in Fig. 9. AND gate B_1 yields an increment $\Delta S = +1$ when both signs of numbers y and S are 0 (i.e., positive numbers) and when there is a carry of 1 to the sign digit in summing y and S by the $\sum$ adder. An increment $\Delta S = +1$ is obtained in the form of a pulse on line a at the application of pulse P_{sd} to the sign digit. Gate B_2 yields a negative increment $\Delta S = -1$ when numbers y and S have sign digit 1 (i.e., negative numbers) and when there is no

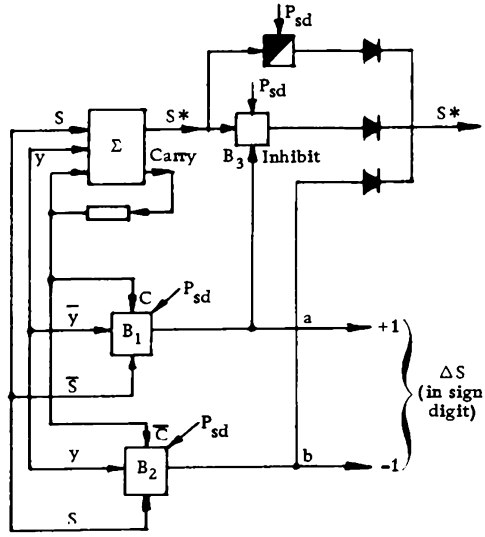


Fig. 9. Generation of ΔS increments by the ternary method (second method).

carry to the sign digit during summation of y and S. The sign of the new number $S^* = S \pm y$, obtained as the result of summation, is determined by gates B_2 and B_3. Gate B_2 delivers a 1 to the sign digit of number S^* when y and S are negative and when there is no carry. Gate B_3 inhibits the passage of the sign code of number S^* from the adder when y and S are positive and where there is a carry (i.e., sign of number S is 0).

Below are presented examples of adding number codes y and S to illustrate the above rules.

Example 1. Numbers y and S have different signs:

	1 11 carry	
a) $S = +2$;	[0]0010	
$y = -1$;	[1]1111	
$+1$	[0]0001	$\Delta S = 0$;
b) $S = +1$;	[0]0001	
$y = -12$;	[1]0100	
-11	[1]0101	$\Delta S = 0$;
c) $S = -8$;	[1]1000	
$y = +7$;	[0]0111	
-1	[1]1111	(−1) $\Delta S = 0$;
d) $S = +15$;	[0]1111	
$y = -8$;	[1]1000	
$+7$	[0]0111	$\Delta S = 0$.

In all of these examples the sign code of the sum thus obtained is correct and the increment $\Delta S = 0$.

Example 2. Both numbers y and S are negative.

a) $S = -8$;	[1]1000
$y = -9$;	[1]0111
-17	[0]1111

There is no carry to the sign digit; $\Delta S = -1$; instead of [0], the sign code of the sum must be [1].

b) $S = -16$; [1]0000

$y = -8$; [1]1000

-24 [0]1000

There is no carry to the sign digit; $\Delta S = -1$; instead of [0], the sign code of the sum must be [1].

1 carry

c) $S = -8$; [1]1000

$y = -1$; [1]1111

-9 [1]0111

There is a carry to the sign digit; $\Delta S = 0$; the sign code of the sum is correct.

1 11 carry

d) $S = -9$; [1]0111

$y = -2$; [1]1110

-11 [1]0101

There is a carry to the sign digit; $\Delta S = 0$; the sign code of the sum is correct.

6.3 An Example

The ternary method of coding increments permits us to obtain a more accurate computation result than in the presence of fluctuations of 1's (+1) or 0's (—1).

As an example, the values of sin x were calculated by the ternary method of representation of increments and shown in Table 5. The maximum capacity of register y and S at which overflow begins to occur was taken to be 128. The maximum value of the sine and cosine is also taken to be approximately 128.

With the chosen scales, it is necessary to make $(\frac{\pi}{2}) \times 128 =$ 201 steps in order to obtain $\sin 90^\circ = 1$. Herein the maximum value of the sine is 128 units. It is seen from the result of the computations that the error in calculating the sine and cosine for an 8-bit register capacity is 0.5-1% when the ternary method is used, whereas the error from using the binary method is almost twice as large.

TABLE 5

Computation of Sine x in the Ternary Method of Coding Increments

Step No.	$y'' = \sin x$	S_1	ΔS_1	$y' = \cos x$	S_2	ΔS_2
	Initial Conditions					
	0	127	0	127	0.000	0
1	0	127	0	127	0 + 127 = 127	+1
2	0	127	0	127	127 + 127 = 254 - 128 = 126	+1
3	1	127 + 1 = 128 - 128 = 0	+1	127 - 1 = 126	126 + 126 = 252 - 128 = 124	+1
4	1 + 1 = 2	0 + 2 = 2	0	126	126 + 124 = 250 - 128 = 122	+1
5	2 + 1 = 3	2 + 3 = 5	0	126	126 + 122 = 248 - 128 = 120	+1
6	3 + 1 = 4	5 + 4 = 9	0	126	126 + 120 = 246 - 128 = 118	+1
7	4 + 1 = 5	9 + 5 = 14	0	126	126 + 118 = 224 - 128 = 116	+1
8	5 + 1 = 6	14 + 6 = 20	0	126	126 + 116 = 242 - 128 = 114	+1
9	6 + 1 = 7	20 + 7 = 27	0	126	114 + 126 = 240 - 128 = 112	+1
10	7 + 1 = 8	27 + 8 = 35	0	126	112 + 126 = 238 - 128 = 110	+1
11	8 + 1 = 9	35 + 9 = 44	0	126	110 + 127 = 236 - 128 = 108	+1
12	9 + 1 = 10	44 + 10 = 54	0	126	108 + 126 = 234 - 128 = 106	+1
13	10 + 1 = 11	54 + 11 = 65	0	126	106 + 126 = 232 - 128 = 104	+1
14	11 + 1 = 12	65 + 12 = 77	0	126	104 + 126 = 230 - 128 = 102	+1
15	12 + 1 = 13	77 + 13 = 90	0	126	102 + 126 = 228 - 128 = 100	+1
		. . . . . etc.				
96	84 + 1 = 85	100 + 85 = 185 - 128 = 57	+1	94 - 1 = 93	93 + 88 = 181 - 128 = 53	+1
97	85 + 1 = 86	86 + 57 = 143 - 128 = 15	+1	93 - 1 = 92	92 + 53 = 145 - 128 = 17	+1
98	86 + 1 = 87	87 + 15 = 102	0	92	92 + 17 = 109	0
99	87 + 1 = 88	102 + 88 = 190 - 128 = 62	+1	92 - 1 = 91	109 + 91 = 200 - 128 = 72	+1
100	88 + 1 = 89	89 + 63 = 151 - 128 = 23	+1	91 - 1 = 90	90 + 72 = 162 - 128 = 34	+1
101	89 + 1 = 90	90 + 23 = 113	0	90	90 + 34 = 124	0
		. . . . .	. . . . .	. . etc.		
200	127	127 + 48 = 175 - 128 = 47	+1	2	74 + 2 = 76	0
201	127	127 + 47 = 174 - 128 = 46	+1	2 - 1 = 1	76 + 1 = 77	0
202	127	127 + 46 = 173 - 128 = 45	+1	1 - 1 = 0	77 + 0 = 77	0

7. SERIAL INTEGRATORS USING A MAGNETIC-DRUM MEMORY

In describing the digital integrator in Sec. 2, each integrator was assumed to have y and S counter-registers using vacuum tubes. However, this would require a prohibitively large amount of electronic equipment for a reasonable number of integrators.

The use of a magnetic drum memory permits integrations in succession by means of one adder when the memory stores the integrand y, integral S, and the integral increment ΔS (integrator outputs).

7.1 S and y Tracks

Instead of using registers, the surface of a magnetic drum is used to store the number codes of y, S and ΔS of all integrators. Each register of an integrator is a sector of a track on the magnetic drum. Assume that the read and write heads of the magnetic drum are diametrically opposed (see Fig. 10). On one half of the drum surface are written the initial data for each integrator in sectors $1'$, $2'$, $3'$, etc. These initial data are read by head a and transferred to the adder. Simultaneous readings of y and S numbers are achieved by using two parallel tracks. As pairs of digits for an integrator are read sequentially, the result of digit-by-digit summation for the integrator is simultaneously written on the other half of the drum surface in sectors 1, 2, 3, etc. by means of head b. Each integration step corresponds to one half of the drum circumference; the number written in any sector on the drum is replaced by a new number after each step. Those numbers that must remain unchanged in the process of computation are rewritten by closing gate B to connect the read and write heads; thereby, the pulses read by head a in sector $1'$ are immediately recorded in sector 1. In this example, the possible number of integrators is limited by the number of sectors on one half of the drum circumference, which is the arc length l of the drum surface between heads a and b.

The effective length l may be almost doubled if the distance between the read and write heads is so changed that writing and reading are performed on adjacent sectors of the same track or even in the same sector. The length of arc of each sector is chosen to accommodate the entire number. The leading digit of each number is reserved for writing the so-called initial pulse H, which serves for control of the computer operation. The trailing digit of the number, after the sign digit, is used for storing the overflow sign in the S_0 register. Thus, the entire number has $n+3$ digits, where n is the number of number digits. Each digit of a number is recorded at a strictly defined location on the drum surface by means of a clock pulse track. For write and read synchronization, the pulse arriving at the read or write head is first stored in a flipflop. This flipflop controls the gate to pass

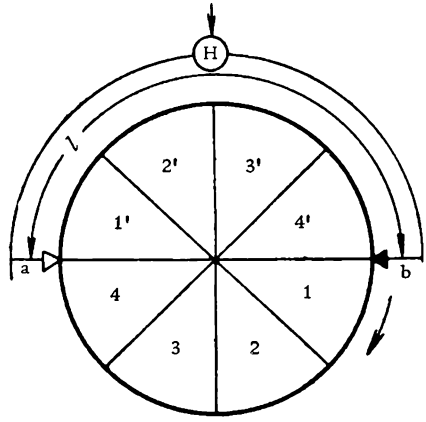


Fig. 10. Position of read and write heads on a magnetic drum.

or not to pass the clock pulses to the write head. Since the clock pulses are spaced at constant and fixed intervals, the number digits are recorded at the same intervals.

For example, if the clock pulses have a spacing of 0.2 mm, the writing of a number with 5 decimal digits (16 binary digits) requires that each sector have an arc of length $(16 + 3)\ 0.2 = 3.8$ mm.

The basic diagram of a serial digital integrating computer is shown in Fig. 11. The principal storage element is a magnetic drum driven by a d-c or a-c motor which rotates at a constant speed (3000-6000 rpm). There are 6 parallel tracks on the surface of the drum. Each track has one read and one write head (with the exception of the clock pulse track CP, which has a read head only). The S and y tracks store the full length of integrands y and integrals S corresponding to registers 2 and 3 in the diagram in Fig. 2.

7.2 ΔS and Address Tracks

The integral increment ΔS (in the form of overflow pulses at the integrator output) is written on a separate ΔS track. In addition, there are four other tracks, y, S, A_x and A_y (see next paragraph); the write and read heads on each of the four tracks have a relative displacement of 180°. The write and read heads on the ΔS track have a displacement slightly less than the length of arc of one sector so that the overflow pulses from all integrators are written and rewritten within the limits of one sector.

The above mentioned A_x and A_y tracks are address tracks on which the so-called address codes of A_x and A_y are recorded. As will be described later, A_x and A_y addresses specify interconnection of the integrators. This connection is achieved by selecting from track ΔS the required overflow pulse and then transferring

it to input A_x or A_y of each integrator. As a result different A_x and A_y codes, depending on the equation to be solved, are written in each sector of the A_x and A_y tracks. They remain unchanged during computation of the problem and are continuously rewritten on the other half of the drum surface by the connection between the read and write heads.

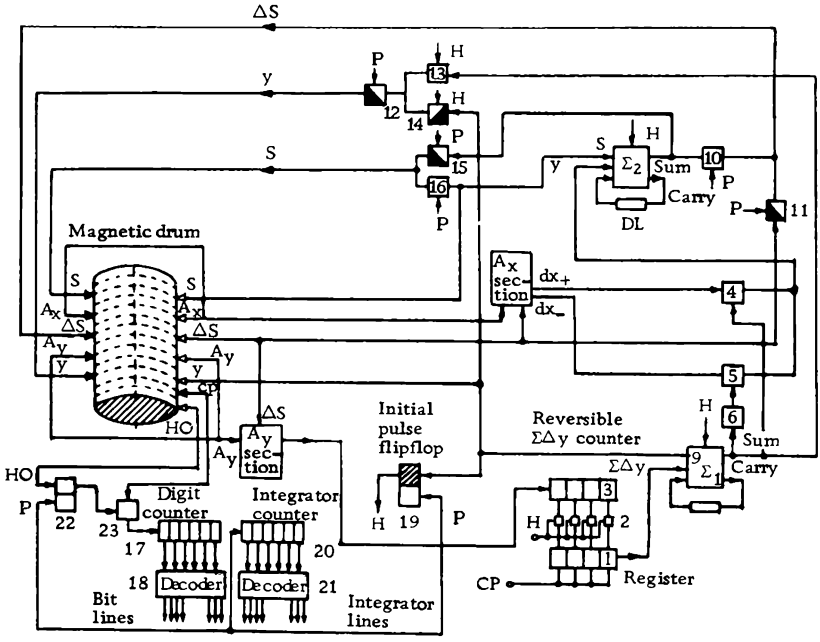


Fig. 11. Basic diagram of serial digital integrating computer with magnetic drum.

7.3 Tracks Being Delay Lines

Each of the tracks y, S, A_x and A_y is in effect a delay line on a magnetic drum. The numbers written on these tracks circulate in the form of a definite sequence of pulses. The circulation frequency is determined by the clock pulse frequency, which in turn depends on the drum speed. The length l of tracks y, S, A_x and A_y is identical and almost equal to the length of the entire drum circumference. It embraces all sectors of the magnetic drum which participate in the computation. The length of a delay line for overflows ΔS is shorter and approximately equal to the length of the arc of one sector. For example, if 60 sixteen-digit numbers for 60 integrators are written on the drum surface and the drum speed is so chosen to give a clock pulse period of 10 μsec, the time length l of the delay line for tracks y, S, A_x and A_y is $60 \times 16 \times 10 = 9600$ μsec $= 9.6$ msec and that for track ΔS is only $15 \times 10 = 150$ μsec.

Magnetic-drum delay lines may be replaced by other types of delay lines (e.g., magnetostrictive, quartz, etc.). The operating principle of the integrating computer remains the same, but the operating speed may be considerably increased. This can in turn decrease the solution time or increase the number of problems to be solved for a given time period.

7.4 Integration Operations

The process of integration, as was stated in Sec. 5, consists of two summation operations for each integration step:

$$y_0 \pm \sum \Delta y = y \text{ (adder } \sum_1\text{)};$$
$$S_0 \pm y = S \text{ (adder } \sum_2\text{)},$$

where y_0, S_0 and y, S are the initial and final values of the integrand and the integral for each step. The integration process begins with the summation of the overflow pulses from the outputs of all integrators; these outputs must be connected to the inputs Δy of the integrators according to the setup for solving the equation. It permits at input Δy of an integrator the application of increments from the outputs of a number of previous integrators.

The execution of summation operation $\sum \Delta y$ is shifted in time by one sector. In other words, the summation of $\sum \Delta y$ for use in the integration of the present sector is performed in the previous sector. Therefore, the read and write heads of ΔS are shifted in space by one sector backward relative to the y and S read heads.

The overflow pulses written in a sector of track ΔS are read and transferred to the Δy or Δx input of the integrator. Selection of the required overflow pulses ΔS for the Δy or Δx input is achieved by the coincidence circuit in the A_x and A_y blocks (Fig. 11) according to the A_x or A_y address code.

As will be shown below, the overflow pluses ΔS from all integrators are written in each sector of the ΔS track in numerical sequence. In order to connect the onput of one integrator with any input of the next integrator, a code of 1 or 0 (representing the presence or absence of a pulse) is written on address tracks A_x and A_y. The address code in each sector is a sequence of 1's and 0's so that the ΔS output of any previous integrator is or is not selected and applied to the input of the integrator by use of a coincidence circuit.

The ΔS increments chosen by the circuit in the Ai block are first summed in the reversible counter $\sum \Delta y$, which sums code 1 as the increment +1 and code 0 as the increment -1. The number in the counter is then transferred to the shift register.

The reversible counter and the shift register usually have three bits plus a sign bit so that increments from seven different

channels may be applied to the Δy input of the integrator. Shift of the $\sum \Delta y$ number and its transfer in the form of a sequence of pulses to the $\sum_1$ adder input is achieved by clock pulses.

The basic diagram of a three-digit $\sum \Delta y$ counter with shift register is shown in Fig. 12. The $\sum \Delta y$ number digits are stored in the counter flipflops and transferred to the shift register in parallel via coincidence circuits $B_1 - B_4$ by a clock pulse applied at the beginning of addition of $\sum \Delta y$ and y. After the transfer, the counter is cleared by a pulse applied to the counter clear bus. During the transfer of number $\sum \Delta y$ to the shift register, the counter sign flipflop closes (nonconducts) gate B_0 when the number $\sum \Delta y$ in the counter is positive and opens (or conducts) the gate B_0 when the number $\sum \Delta y$ is negative. When $\sum \Delta y$ is negative, the application of n clock pulses to the shift bus makes the $\sum \Delta y$ register give the 1's complement of number $\sum \Delta y$ with the number of digits equal to n digits of number y with which $\sum \Delta y$ is added. For example, the numbers $+6$ and -6 in the $\sum \Delta y$ counter are presented at the register output as follows:

$+6$ code: [0] 000. . .00110

-6 code: [0] 11. . .11010

Simultaneously with the number $\sum \Delta y$ there arrives at the input of the $\sum_1$ adder a sequence of pulses representing the number y read from the magnetic drum. Single-bit adder $\sum_1$ succesively adds or subtracts numbers y and $\sum \Delta y$, depending on the sign of the number in the $\sum \Delta y$ counter. A positive number $\sum \Delta y$ has code 1 in the sign location of the $\sum \Delta y$ counter and is expressed in direct code while a negative number has code 0 and is expressed in complemented code.

The summed result, the new value of $y = y_0 \pm \sum \Delta y$ at the output of adder $\sum_1$, is transferred to the input of adder $\sum_2$ through gates 4 and 5 (refer to Fig. 11). At the same time, the output of adder $\sum_1$ is connected through gate 13 to the write head after the writing of the initial pulse in the y register. Initial pulse H serves as the signal for beginning of addition in the adders. It may first be written in the y register in any digit position. This

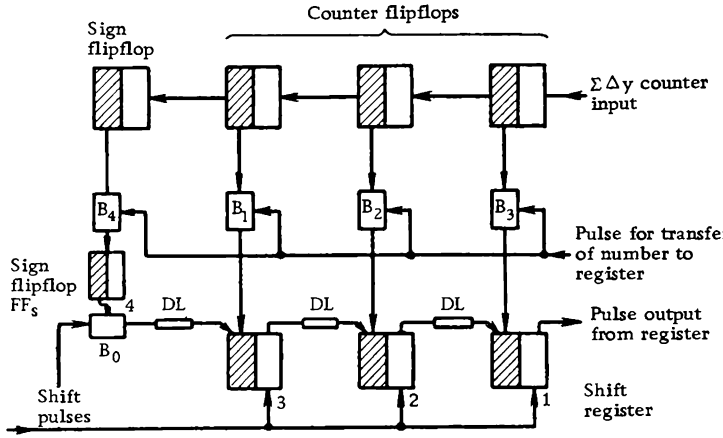


Fig. 12. Three-digit reversible counter.

is determined by the chosen scales for the numbers $\sum \Delta y$ and y. For example, if the initial pulse is written in the third digit of the number code for y, the least significant digit of the number $\sum \Delta y$ is added with the third significant digit of the number y (i.e., there is a two-digit shift to the right between the augend and addend in accordance with the chosen scale).

The initial pulse stored in the y register turns flipflop 19 on upon application to one of its inputs. This flipflow remains on during the time the number y arrives from the y read head. Only after the passage of pulse P, the last digit of number y, is flipflop 19 turned off.

The initial pulse controls gates 13 and 14 with the output of flipflop 19. Gate 13 is an anticoincidence circuit which allows passage (to the write head from the output of adder $\sum_1$) of all pulses except the initial pulse. Gate 14 is a coincidence circuit and passes only the initial pulse from the y channel for rewrite.

The new value of y from the $\sum_1$ adder output reaches the $\sum_2$ adder input simultaneously with the number S_0 from read head S of the magnetic drum.

The A_x block for selection of overflow pulses according to the A_x address controls gates 4 and 5 through which the number y reaches the $\sum_2$ adder. Depending on the code 1 or 0 at the output of the coincidence circuit, gate 4 or 5 is open. If increment Δx is positive (Δx_+), gate 4 is open and the number y arrives at the input of the $\sum_2$ adder (i.e., $S_0 + y$ is performed). If increment

Δx has a negative sign (Δx_-), gate 5 is open and the 1's complement* of the number y arrives at the input of the $\sum_2$ adder (i.e., S_0 and $-y$ is performed). The result of summation of S_0 and y is written on track y of the magnetic drum by connecting the output of the $\sum_2$ adder through anticoincidence gate 15 to the write head of track S. Gate 15 remains open except during the last clock pulse P (the last digit of the S code). During this period, the code of the last digit of the number S is rewritten through the open gate (coincidence circuit) 16.

During the last clock pulse P the overflow pulse from the output of adder $\sum_2$ is written through coincidence circuit 10 into the ΔS overflow-pulse storage line. The overflow pulses from all the previous integrators are rewritten into the ΔS line through anticoincidence gate 11, which is closed only during the last clock pulse P of the given sector. Anticoincidence gate 12 does not allow the passage of the last digit of number y to the record head, so that the last digit of y is always 0.

7.5 Clock and Control Pulses

Control of all computer operations is achieved by means of clock pulses uniformly recorded on the CP track of the magnetic drum. Binary counter 17 counts the clock pulses. In order that a specific digit location on the drum correspond to the same counter reading regardless of the moment of turn-on of the circuit, a reference pulse is recorded at a definite location on track *HO*. When the *HO* head reads the reference pulse, flipflop 22 is turned on and gate 23 is opened, admitting the clock pulses to the digit counter. After passage of the last clock to the counter, flipflop 22 is turned off. The digit counter (with 4-5 bits) is connected to decoder 18 having 16 (or more) output lines, each corresponding to a specific digit time in the order of the number digits in the y and S registers.

The drum sectors (integrators) are counted by integrator counter 20 which has 5 or 6 binary digits, depending on the total number of integrators in the computer.

At the input of the integrator counter pulses arrive from the last output line of the digit counter decoder (that is, the carry pulse from the digit counter). The integrator counter** has a decoder having, for example, 64 output lines corresponding to 64 integrators of the computer. These output lines control switching of the circuits. If the clock repetition frequency of the computer is 100 kc, switching at the output lines of the digit counter decoder is also at a frequency of 100 kc. Switching of the output lines at

*The 1's complement of y differs from the 2's complement of y by one in the least significant digit, which may be disregarded in the S_0 code.

**Instead of the integrator counter with decoders, shift registers or ring counters may be used (see Chapter III).

the integrator counter decoder (with 16-digit y and S) is at a frequency of only $\frac{100}{16} = 6.25$ kc.

8. INTERCONNECTION OF INTEGRATORS

8.1 Overflow Line

The overflow pulses ΔS from the integrators outputs are stored on the ΔS track. The length of overflow line on the ΔS track is one digit shorter than the number of digits that can be written within one sector. In each sector of the magnetic drum there can be $n + 3$ pulses, where n is the number of number digits. Of the three digits, one digit is the sign digit, another digit denotes the start pulse, and the third digit (located at the last digit position) records the overflow pulse. Only $n + 2$ digits may be written in the ΔS line. The overflow pulses stored in the ΔS line are rewritten in all the remaining sectors of the magnetic drum. Due to the fact that the read and write heads of the ΔS line are displaced by one digit, the overflow pulses are each time displaced by one digit during rewriting in the next sector. For example, let us assume that each track on the magnetic drum is divided into six sectors and that six digits may be written in each sector (i.e., a maximum of six overflow pulse codes). Assume (Fig. 13) that the overflow pulse of each integrator (sector) is written during the last clock of each sector at the points u_1, u_2, u_3, u_4, u_5, u_6. This means that, because of the specified distance between the read and write heads, with location of the read head at point u_1 of sector 1 the overflow pulse of the first integrator is written in the second sector at the location denoted by the number 1. This pulse will be rewritten in the third sector at a location further displaced by one digit. When the read head is at u_2 the new overflow pulse of the second integrator (sector), denoted by the number 2, is written. As the drum rotates, the overflow pulses from the previous integrators are written in definite sequence according to numerical order of the integrators (sectors) and the new overflow pulse from each integrator follows. When the drum completes one and continues to another revolution, instead of pulse 1 of the first integrator in sector 1, a new overflow pulse from integrator 1 is written in sector 1. The new overflow pulses are written in other sectors 1 at the same position as described before. To write a new overflow pulse it is necessary that gate B_1 be closed only during the last digit of each sector (i.e., at points $u_1, u_2, \ldots, u_6$). Gate B_2, through which the overflow pulse arrives from the integrator output, must be open only at points $u_1 - u_6$. In order to write the overflow pulses from all integrators in one sector on track ΔS, the number of overflow pulses must not be larger than the number of digits which may be written in one sector.

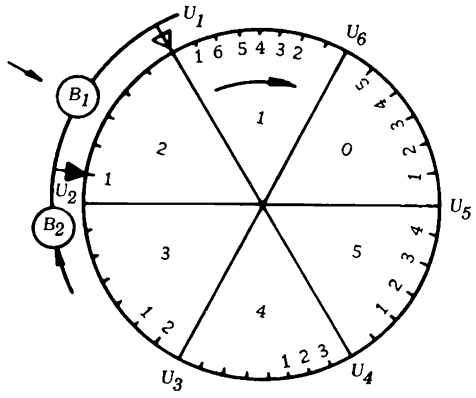


Fig. 13. Operational diagram of overflow line.

Table 6 below lists the overflow pulses for the above discussed example with six integrators (sectors) and with six digits in each sector.

As is seen from the table, during the second revolution of the drum (second step of integration) new values of overflow pulses are written (1 new, 2 new, etc.) instead of the old values.

In the diagram in Fig. 11 the read head of track ΔS is connected with the write head through gate 11, which functions as gate B_1 (Fig. 13). Gate 11, an anticoincidence circuit, remains open in rewriting the overflow pulses and closes only during the last clock

Table 6

Status of Overflow Pulses

Digit No.	1st revolution of drum						2nd revolution of drum			
	Section No.									
	1	2	3	4	5	6	1	2	3	4
1st digit (lowest)						1	2	3	4	5
2nd digit					1	2	3	4	5	6
3rd digit				1	2	3	4	5	6	1 new
4th digit			1	2	3	4	5	6	1 new	2 new
5th digit		1	2	3	4	5	6	1 new	2 new	3 new
6th digit (last)	1	2	3	4	5	6	1 new	2 new	3 new	4 new

pulse P in each sector. A new overflow pulse is transferred from the output of the $\sum_2$ adder integrator to the write head of track ΔS through gate 10, which is open only during the last pulse P.

8.2 Addressing Circuits

Selection of the overflow pulses stored in the ΔS line is actuated by the two addressing circuits Ax and Ay. Interconnection of the integrators is achieved by means of these circuits, which during each integration step select the overflow pulse ΔS of any of the preceding integrators and apply it to input Δy or Δx of the next integrator (see Chapter V, Sections 8, 9).

For example, let us discuss how the overflow pulse codes from line ΔS are selected and applied to input Δy of the next integrator according to the Ay code written on the Ay track. Figure 14a shows the addressing circuit Ay in which code 1 or code 0 from the read head of ΔS line is converted to pulses. These pulses pass through gate 1 or 2 and reach the reversible counter $\sum \Delta y$ only upon their coincidence with the pulses written as code 1 on track Ay. The pulses on bus a are applied to the add input of the counter and those on bus b to the subtract input. As shown in Fig. 14a, increment pulses of the integrand may be applied directly to the counter from external input units.

The Ax addressing circuit for selecting overflow pulses for input Δx of the next integrator is shown in Fig. 14b. The overflow pulses stored in the ΔS line are transferred to bus a or b and stored in flipflop FF_1, which has one input for code 1 and the other for code 0.

Upon the arrival of initial pulse H of an integrator the code stored in flipflop FF_1 is transferred to flipflop FF_2 via gates 3 and 4. Flipflop FF_2 controls gates 5 and 6 through which the number y or its 1's complement passes from the $\sum_1$ adder to the $\sum_2$ adder. The FF_2 flipflop outputs denoted by Δx_+ and Δx_- correspond respectively to a positive and a negative increment at the Δx input of the integrator. The 1's complement of y is obtained from its direct code by means of gate 7. Gate 7 grounds clock pulses CP arriving through gate 8 when the given digit of y is 1 and almost all the voltage flows to a large resistor R. Conversely, clock pulses pass gate 8 on the way to the $\sum_2$ adder when the given digit of y is 0 because gate 7 is closed. Thus, we obtain the 1's complement of number y. The described method of obtaining the 1's complement of y may also be applied if an anticoincidence circuit is used instead of gate 7.

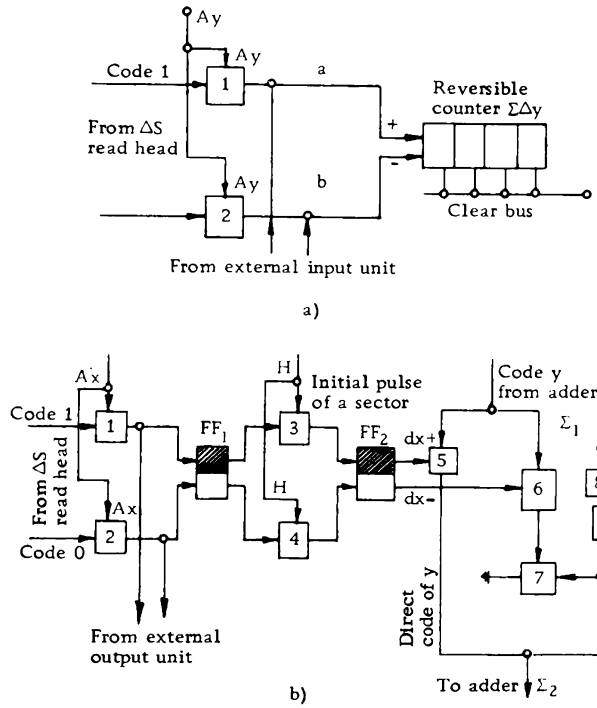


Fig. 14. Addressing circuits for overflow pulses.

9. INSERTION OF INITIAL CONDITIONS. PROBLEM SWITCHING

Before beginning the solution of any problem the initial values of all integrands must be inserted into the y registers. In addition, address codes must be written on tracks Ax and Ay for interconnection of the integrators. Thus, before solving a problem, it is necessary to enter the coded information in all sectors on tracks y, Ax and Ay of the magnetic drum. Before computation begins, all data on tracks S and ΔS must be erased or written only as 0's.

9.1 Problem Switching

Problem switching in the computer is simple if the tracks for y, Ax and Ay can be switched from one channel to another. When a magnetic drum storage is used, the initial conditions and program codes for various problems may be written on separate groups of tracks (Fig. 15), one group of which is selected upon

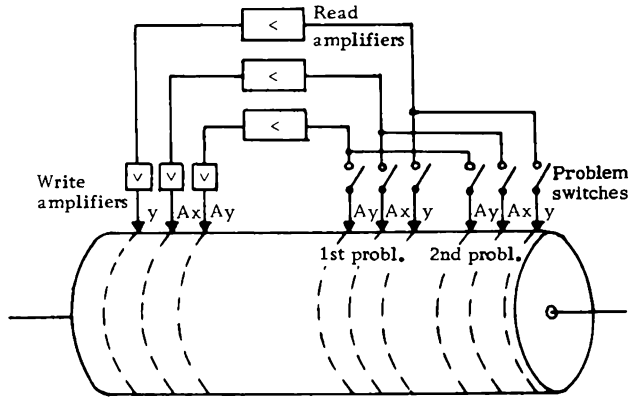


Fig. 15. Switching of initial conditions on drum tracks.

switching to the problem to be solved. This requires an additional three read amplifiers for this data.

9.2 Inital Solution Time

In solution of a problem by the integration method, a certain initial time is required to reach a given point on the curve under integration. This initial solution time, or "build-up time," may be reduced by selection of initial conditions lying as close to the given point as possible. It is evident that the initial solution time for a problem may be reduced by one-half if the initial conditions are chosen at a value lying in the middle of the given range of values of the independent variables of the equation.

With a small number of independent variables in the problem to be solved, it is possible to reduce the initial solution time of the problem by a factor of 4-8 or larger in comparison with the initial time when initial conditions are given at the initial point of the curve under integration.

Let us assume that it is required to compute the value of the following function U with three independent variables x, y and z:

$$U = f(x, y, z), \tag{23}$$

Let us choose for each of the independent variables x, y and z two points (x_1 and x_2, y_1 and y_2, z_1 and z_2) lying in the middle of the first and second half of the possible range of values of these variables (Fig. 16). If the chosen point of the curve under integration lies in the first half of the range of values in variables x, y and z, with the inserted initial value of function $U_1 = (x_1, y_1, z_1)$, the initial solution time will be decreased by a factor of 4. This also occurs for the second half of the range if we take the following function as the initial condition for the function,

$$U_2 = f(x_2, y_2, z_2). \tag{24}$$

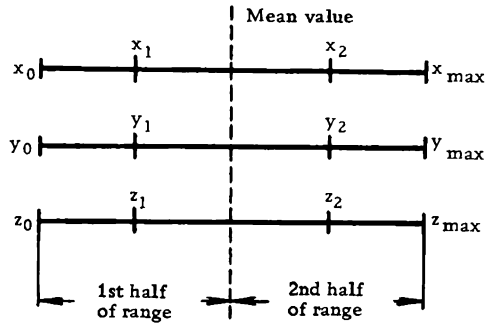


Fig. 16. Combinations of initial conditions.

In general, for any values of independent variables x, y and z, the initial solution time may be reduced by a factor of 4 if one of eight possible sets of initial values is switched according to the value of x, y and z.

In effect, the number of possible combinations for the three independent variables, each having two values, is $2^3 = 8$. Depending on the half of the range (first or second) in which each of the three independent variables x, y and z lies, we have the following eight sets of initial values for function U:

$$\left.\begin{aligned} U_1 &= f(x_1, y_1, z_1), \\ U_2 &= f(x_2, y_2, z_2), \\ U_3 &= f(x_1, y_1, z_2), \\ U_4 &= f(x_1, y_2, z_2), \\ U_5 &= f(x_2, y_2, z_1), \\ U_6 &= f(x_2, y_1, z_2), \\ U_7 &= f(x_1, y_2, z_1), \\ U_8 &= f(x_2, y_1, z_1). \end{aligned}\right\} \qquad (25)$$

9.3 Insertion of Initial Conditions

Automatic insertion of initial conditions may be achieved by means of diode-transformer matrices. Let us illustrate this by an example. Assume that function $U_1 = f(x, y, z, u)$, a function of 4 independent variables x, y, z, u, is an algebraic or a trigonometric function. Further assume that two initial values are given for each of the four independent variables (this reduces the solution time by a factor of 4 in comparison with the zero initial values).

Then, for the 16 possible combinations of four variables there are 16 possible initial conditions. These initial conditions are automatically selected by means of the matrices (Fig. 17), in which lines x_1, x_2, y_1, y_2, z_1, z_2, u_1 and u_2 are controlled by the

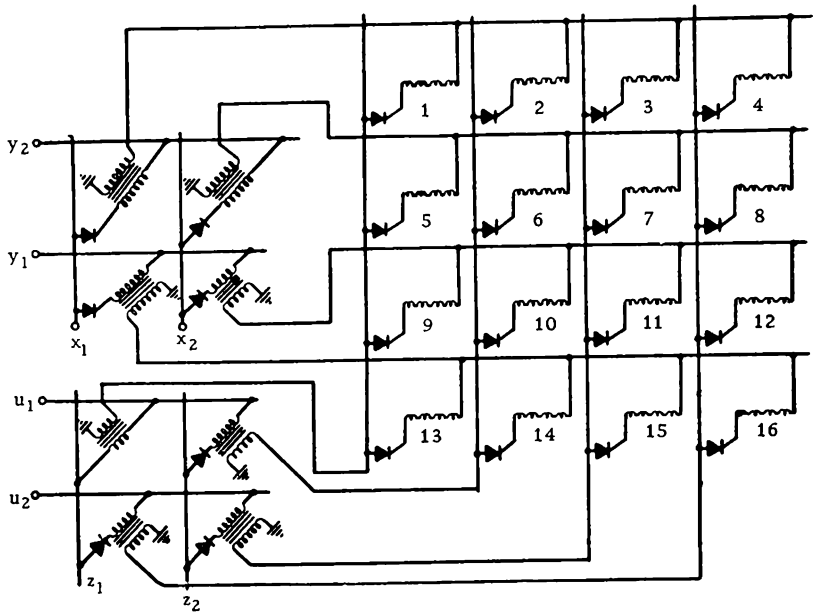


Fig. 17. Selection of initial conditions.

input values of x, y, z and u in the following manner. When the input variables are less than half of their maximum values, lines x_1, y_1, z_1 and u_1 are connected; and when they are larger than half of their maximum values, lines x_2, y_2, z_2 and u_2 are connected. Each of the two diode-transformer matrices generates four outputs; there are four different combinations of x_1, x_2, y_1 and y_2 in the first matrix and four combinations of z_1, z_2, u_1 and u_2 in the second matrix. The eight output lines of these two matrices drive a third diode-transformer matrix having four horizontal and four vertical lines. At each point of intersection of these lines there is connected a diode in series with the windings of 18 transformers (for an 18-digit number) by means of which the 16 initial conditions for all the integrators are stored (permanent memory, see Chapter V, Section 4). These transformers must have $16n$ windings, where n is the number of integrators. Such a permanent memory for the initial condition storage and automatic selection of these conditions in accordance with the given parameters permits a considerable decrease in the time required for generation of initial data in the machine.

10. SERVO INTEGRATOR

Analogous to an ordinary servomechanism, the servo integrator serves as a type of error-nulling device in the comparison of two

(or more) numbers. The difference Δ of two compared numbers, $A - B = \Delta$, (or the amount of error or unbalance between them) is applied to the input dy. The error Δ is either in the form of pulses or in the form of a number stored in the $\sum\Delta y$ counter.

With each step of integration, there occurs at the output dS of the integrator a sequence of overflow pulses depending on the sign of Δ. If Δ is a positive number, the output is a sequence of codes 0, 0, 0, 0, 0, etc., until the error is zero. If Δ is a negative number, the output is a sequence of codes 1, 1, 1, etc., until the error is zero. If Δ is 0, the code sequence at the output is 0, 1, 0, 1, 0, 1, etc.

Thus, with each step of integration the servo integrator generates codes 1 or 0, depending on the sign of the error Δ. The generated pulse codes are fed back to the integrator input. Since these pulses are applied to the $\sum\Delta y$ counter input with an opposite sign (negative feedback), the amount of error decreases with each integration step.

10.1 Operating Principle

The servo integrator operates on the principle that addition of -1 to the largest negative number 0.000...00 initially written in the y register converts it to the largest positive number 1.111...11. Conversely, the addition of +1 to the largest positive number 1.111...11 converts it to the largest negative number 0.000...00. Let us explain the operation of the servo integrator with a few examples.

Consider the servo integrator in Fig. 18. Assume that the error applied to the integrator input is zero. The number 0.00...000 is initially in the y and S registers. Code 1 (e.g., the overflow pulse from a digit counter) is applied to the input dx with each step of integration. The overflow pulses from the dS output of the integrator are fed back to the dy input. The results obtained after each integration step are shown in Table 7. It is seen from the table that after the initial steps the sequence of dS codes at the integrator output is 0,1, 0,1, etc.

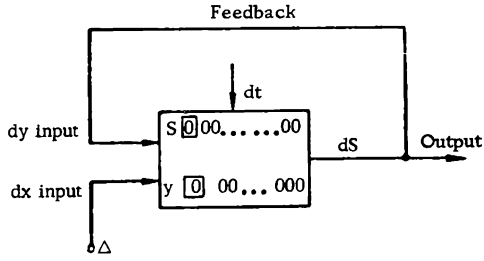


Fig. 18. Servo integrator diagram.

Table 7

Overflow pulses at servo integrator output with zero error Δ at input

Integration Step	Increment Code Δy	$\sum \Delta y$ Register Code	y Register Code	S Register Code	Overflow Pulse Code
1	—	[0]00000	[0]00000	[0]00000	
				[0]00000	
				[0]00000	0
2	[1]11111	[0]00000	[0]00000	[0]00000	
		[1]11111		[1]11111	
		[1]11111		[1]11111	0
3	[1]11111	[1]11111	[1]11111	[1]11111	
		[1]11111		[1]11110	
		[1]11110		[1]11101	1
4	[0]00001	[1]11110	[1]11101	[1]11101	
		[0]00001		[1]11111	
		[1]11111		[1]11110	1
5	[0]00001	[1]11111	[1]11100	[1]11100	
		[0]00001		[0]00000	
		[0]00000		[1]11100	0
6	[1]11111	[0]00000	[1]11100	[1]11100	
		[1]11111		[1]11111	
		[1]11111		[1]11011	1
7	[0]00001	[1]11111	[0]11011	[1]11011	
		[0]00001		[0]00000	
		[0]00000		[1]11011	0
8	[1]11111	[0]00000		[1]11011	
		[1]11111		[1]11111	
		[1]11111		[1]11010	1
				etc.	

Let us now assume that $\Delta = 2$ (in the decimal system); then in decimal notation we obtain:

In first step		$\boxed{0}\,000\,S$
	$+$	$\boxed{0}\,002\,y$
Overflow pulse $dS = 0$		$\boxed{0}\,002\,S$
Second step		$\boxed{0}\,002\,S$
	$+$	$\boxed{0}\,001\,y$
Over pulse $dS = 0$		$\boxed{0}\,003\,S$
Third step		$\boxed{0}\,003\,S$
	$+$	$\boxed{0}\,000\,y$
Overflow pulse $dS = 0$		$\boxed{0}\,003\,S$
Fourth step		$\boxed{0}\,003\,S$
	$+$	$\boxed{1}\,999\,y$
Overflow pulse $dS = 1$		$\boxed{0}\,002\,S$
Fifth step		$\boxed{0}\,002\,S$
	$+$	$\boxed{0}\,000\,y$
Overflow pulse $dS = 0$		$\boxed{0}\,002\,S$, etc.

In this case, there are two extra code 0 at the integrator output compensating the error $\Delta = 2$ at the $\Delta\ y$ input until the overflow pulses at the output attain the sequence of 1, 0, 1, 0, 1, 0, etc.

The servo integrator may be directly used for comparison of two quantities $A - B = \Delta$ if increments dA and dB are applied through two channels to the dy input of the integrator (Fig. 19). Then at the dS output the difference $(A - B)$ is generated in the form of overflow pulses (with sign reversed).

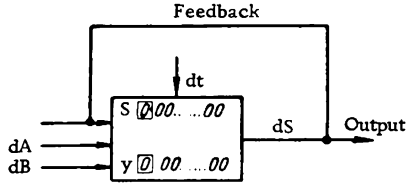


Fig. 19. Comparison of two quantities by use of a servo integrator.

In the general case, with several channels for increments dA, dB, dC, etc., the number of overflow pulses at the integrator output is equal to the algebraic sum of the increments at all its inputs, that is,

$$dS = -(dA + dB + dC + \ldots). \tag{26}$$

This means that if any of the quantities A, B, C introduces at the integrator's dy inputs a pulse (code 1), the integrand y will have the code $\boxed{0}00\ldots001$ and in the feedback circuit there appears an overflow having the code 0. Conversely, if any of the quantites A, B, C introduces at the integrator input an increment in the form of code 0, the integrand y will have the code $\boxed{1}111\ldots11$. Then at the integrator output the overflow will have code 1 and feedback will tend to convert the integrand to code $\boxed{0}00\ldots000$.

10.2 Summing and Operation Integrators

The servo integrator is often used as a summing integrator for solution of various problems. This is necessary, for example, in order to obtain at the dy input of a servo integrator increment pulses which correspond in number to the sum of the increments from several channels. Only one pulse at a time may be applied to the dy input, but the number of such pulses must be equal to the sum of the individual increments.

In the presented examples the feedback loop of the servo integrator joins the input and output of only one integrator. In the solution of complex problems the feedback loop of a servo integrator may include several integrators.

In this case, the servo integrator functions as an operation or decision integrator and is a type of "digital controller." As an example, let us discuss solution of the equation

$$y = \arcsin f(x)$$

or

$$\sin y = f(x), \tag{27}$$

where $f(x)$ is a given function of an independent variable satisfying the condition $-1 < f(x) < +1$. It is required to find the value of y satisfying the given equation for all values of x.

In hooking up the integrators (Fig. 20) y is assumed known. Function $\sin y$ is generated by means of integrators 1 and 2. To the dy input of the servo integrator 3 is applied the difference of increments

$$d(\sin y) - df(x) = d\Delta$$

and the integrand at first has the code $\boxed{0}00\ldots00$. The dS overflow pulses of the servo integrator are fed back to the inputs of integrators 1 and 2 as the increments dy. Then servo integrator 3 will tend to control the dy increments in such manner that the difference $\sin y - f(x) = \Delta$ diminishes to zero. If $\sin y > f(x)$, at the output of the servo integrator there will occur a sequence of codes 0, 0, 0, etc. which decreases $\sin y$. If $\sin y < f(x)$, at the output of integrator 3 there will occur a sequence of codes 1, 1, 1, etc. which increases $\sin y$. Thus, as in an ordinary servomechanism, y is controlled by means of a servo integrator.

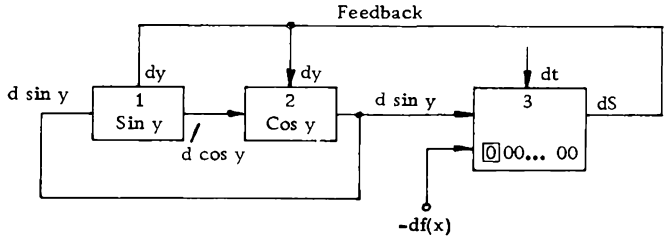


Fig. 20. Solution of $y = \text{arc}\sin f(x)$ by means of a servo integrator.

The rate of generation of the required quantity y depends on the chosen scales for the overflow pulses arriving at the dy input through the feedback loop. In order to increase the rate of generation of quantity y the overflow pulses may arrive at the dy input on a scale greater than one. This is achieved by multiplying them by a factor $k > 1$ in an intermediate scaling integrator (see Sec. 11) or by transferring the increment dS to the second, third or higher digit of the y counter register. This is equivalent to multiplication of dS by a coefficient which is a multiple of two.

In the general case, the feedback loop may include several integrators, and the required scales are chosen during the setup for problem solution.

Servo integrators are widely used for generation of various functions in the solution of a problem. They permit problem solution with fewer integrators.

11. SCALE FACTORS

The limited capacity of the y and S registers of an integrating computer necessitates the establishment of a definite scale for all quantities with which the computer operates. To each quantity in the machine there corresponds a certain scale

$$M = 2^m,$$

where 2 is the radix of binary numbers and m is the power to which the radix 2 must be raised in order to equal M.

The scale M indicates the number by which one unit of the quantity is represented in the machine. For example, if a quantity A is inserted into the y register of the machine with a scale $M = 2^3$, this signifies that one unit of quantity A is represented in the machine in the form of 8 pulses.

11.1 Scale Factor

In subsequent discussion m, the power of two, will be referred to as the scale factor of the scale M. As a rule, all quantities in an integrating computer are represented in binary or decimal numbers. For binary numbers, the scale factor m can only be a number which shows that one unit of a given quantity is represented in the machine in the form of a number 2^m of pulses and the y register must have a capacity of 2^m to store one unit of the given quantity.

11.2 Scale Factor For One Integrator

Let us first establish the scaling relations for a single integrator. Let m_x, m_y and m_s denote the scale factors for the input quantities x and y and the output quantity S of the integrator. If both y and S registers have n binary digits, then, as was established in Sec. 2,

$$dS = \frac{1}{2^n} y dx, \tag{28}$$

If increments dx, dy and dS are represented in the machine with scale factors m_x, m_y and m_s, then instead of y in equation (28) the actual relation between quantities x, y and z is

$$2^{m_s} \cdot dS = \frac{1}{2^n} \cdot 2^{m_y} \cdot y\, 2^{m_x} \cdot dx, \tag{29}$$

or

$$dS = 2^{m_y + m_x - m_s - n} \cdot y \cdot dx. \tag{30}$$

It is desirable that the increment dS in an integrating computer have the same scale as the quantity $y \cdot dx$ (that is, $dS = ydx$). Then, the coefficient of $y \cdot dx$ in equation (30) must be unity, or

$$m_y + m_x - m_s - n = 0. \tag{31}$$

Equation (31) is the basic relation for the scale factors of a single integrator. It establishes the relationship between the number of digits n in the machine registers, the scale factors of input quantities y and x of the integrator and the scale factor for the integral at the integrator output.

The capacity of the y register (i.e., the number n of bits) is determined by two quantites: the scale factor m_y chosen for the integrand and the largest value of the integrand y, which, with the chosen scale, may reach $2^{m_{y\max}}$, where $m_{y_{\max}}$ is the maximum value of the power of two corresponding to

$$y_{\max} = 2^{m_{y\max}} \cdot 2^{m_y}. \tag{32}$$

The chosen number of digits n of the y register must evidently be greater than (or equal to) the sum of $m_{y_{\max}}$ and m_y,

i.e., $$n \geqslant m_{y_{\max}} + m_y. \tag{33}$$

The value of $m_{y_{\max}} + m_y$ is the number of digits required in the y register for storage of the maximum value of y therein.

If, in choosing the scales, the number $m_{y_{\max}}$ is unknown because the maximum of the integrand is unknown, the y register capacity may prove to be inadequate for the chosen scale factor m_y. In this case, after reaching the limit of the register capacity, the integrating computer automatically stops and solution of the problem is interrupted; the scale factor must then be changed.

From equations (31) and (33) we may derive the following relation:

$$m_x - m_S \geqslant m_{y_{\max}} \tag{34}$$

or

$$m_y = m_S - m_x + n \leqslant n - m_{y_{\max}}$$

or

$$m_S - m_x \leqslant m_{y_{\max}}$$

11.3 Scale Factors For Interconnected Integrators

Scales for interconnected integrators are chosen in accordance with the problem-solution hookup. The scaling procedure for interconnected integrators is described below.

Let m_S be the scale factor for the output of the first integrator. Find the scale factor m_S from equation (34) and express it in terms of scale factors m_x and m_y. In similar manner we establish the scales for the second, third, and other integrators.

If the several increments from the outputs of several integrators are applied to the dy input of one integrator, the scale factors for the outputs must be identical. This insures consistency of all the scale factors.

From expression (34) we may write a series of inequalities (one for each of the integrators) which must be simultaneously satisfied:

$$\left.\begin{aligned} m_{x_1} - m_{S_1} &\geqslant m_{y_1\,\max}, \\ m_{x_2} - m_{S_2} &\geqslant m_{y_2\,\max}, \\ &\cdots\cdots\cdots \\ m_{x_k} - m_{S_k} &\geqslant m_{y_k\,\max}. \end{aligned}\right\} \tag{35}$$

The simultaneous solution of these inequalities determines the required scale factors m_x, m_y and m_S for each integrator.

Using equation (29) and the given number of digits n in the y and S registers, we may establish the scale factors for all the remaining variables; or, conversely, from the chosen m_x, m_y, and m_S we may establish the required number of digits of the register.

In both cases it must be kept in mind that a one-bit increase in the independent variable doubles the computation time (number of steps). Thus, an increase in computing accuracy by a factor of 2 is achieved by increasing the computing time; and, conversely, a decrease in computing time is achieved at the expense of a decrease in accuracy.

11.4 Scaling Integrator

In establishing scale factors it is often necessary to multiply the result of integration by a certain constant scale factor K. This is achieved by means of a so-called scaling integrator. If scale factor $K < 1$, we may use a scaling integrator K (Fig. 21a) in which a constant number $K < 1$ is inserted as the integrand. At its output we obtain

$$KdS = Kydx. \tag{36}$$

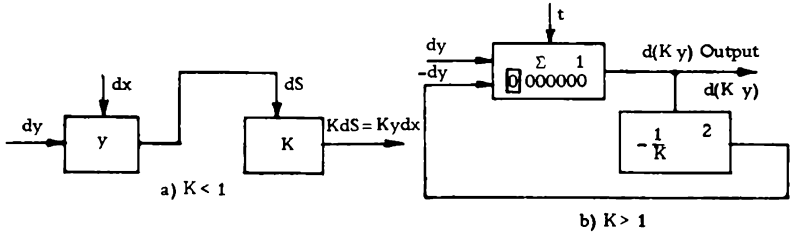


Fig. 21. Scale-factor multiplication diagram.

We may also achieve multiplication of dy or dS by a constant factor $K > 1$. This permits an increase in the number of overflow pulses at the integrator output by a factor K and, in certain cases, an increase in the computing accuracy. The circuit of Fig. 21b

performs the multiplication Kdy by means of a servo integrator 1 and a scaling integrator 2. We assume that a number $-\frac{1}{K}$ is taken as the integrand of integrator 2 and $d(Ky)$ becomes the independent variable which is applied to the input dx. We then obtain at the output of integrator 2 an increment $dS = -\frac{1}{K} d(Ky) = dy$. Servo integrator 1 compares increment dy of the given function with increment $-dy$ from integrator 2 in the feedback loop.

Overflow pulses will be generated at the servo integrator output until the error at the servo integrator input is zero. Herein the number of overflow pulses at the output will be K times greater than at the input (i.e., Kdy), where $K > 1$.

Chapter II

Methods of Solution of Mathematical and Logical Problems

1. SOLUTION OF DIFFERENTIAL EQUATIONS

The methods of solution of mathematical problems in a digital integrating computer are, in principle, analogous to the methods of their solution in electronic analog computers or electromechanical differential analyzers. Just as in analog differential analyzers, the problem to be solved is presented in differential form, most often in the form of differential equations which are solved by means of digital integrators. In setting up diagrams for solution of differential equations, two basic principles are employed: lowering the order of derivatives and using feedback. Lowering the order of derivatives consists in the use of integrators in obtaining the lower-order derivatives from higher-order derivatives. Another possible method is to obtain the higher-order derivatives as the result of differentiation of lower-order derivatives. Since the differentiation operation results in considerably lower accuracy, this method is not used for the solution of differential equations.

As an example of lowering the order of a derivative, let us discuss solution of the equation (Fig. 22a)

$$\frac{d^2y}{dx^2} = f(x). \tag{1}$$

At the output of the function unit 1 we obtain the desired function f of independent variable x. After integration by integrator 2, we obtain the first derivative:

$$\frac{dy}{dx} = \int f(x)\,dx. \tag{2}$$

When the value of $\frac{dy}{dx}$ generated at the output of the integrator 2 is applied to the input of the integrator 3, we obtain

$$y = \int \frac{dy}{dx}\,dx.$$

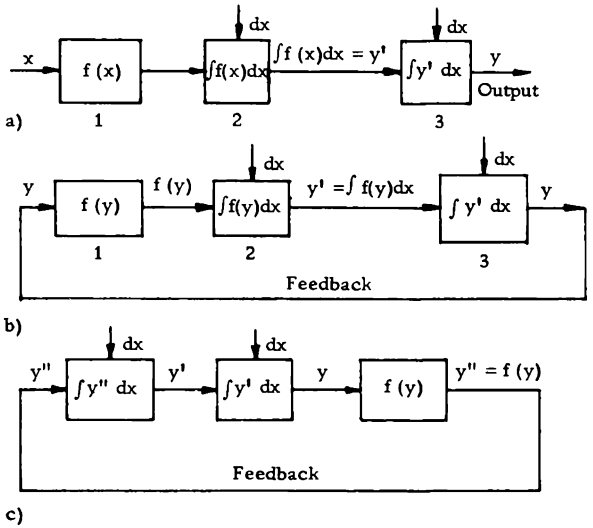


Fig. 22. Diagrams for solution of equations:
(a) $y'' = f(x)$; (b) $y'' = f(y)$; (c) variant of solution $y'' = f(y)$.

Thus, solution of the equation requires two integrators and one function generator.

In the above example, there is no feedback from output to input. Let us now consider an example where the use of feedback is necessary. Assume that the following equation is given:

$$\frac{d^2y}{dx^2} = f(y). \tag{3}$$

Since the function of unknown y is on the right side of equation 3, y can not be determined directly by the method of lowering the order of the derivative by integration alone.

Assume that y is known. Then, by means of function unit 1, we find $f(y)$. By using $f(y)$ as the integrand in integrator 2 (Fig. 22b), we obtain at the output of this integrator

$$S = \int f(y)\,dx. \tag{4}$$

By using S as the integrand in integrator 3, we find

$$S_1 = \int S dx = \iint f(y)\,dx. \tag{5}$$

If the value of S_1 at the output of integrator 3 is fed back to the input of the function generator 1, the value of y is in agreement with the given equation (3).

It is not obligatory to assume that y is known when setting up the diagram. For example, assume that $\frac{d^2y}{dx^2}$ is known. Then, successive integrations by means of integrators 1 and 2 (Fig. 22c) first give

$$\frac{dy}{dx} = \int \frac{d^2y}{dx^2}\, dx, \tag{6}$$

and then give

$$y = \int \frac{dy}{dx}\, dx. \tag{7}$$

Function y is obtained by applying y to function unit 3. If now the output of function generator 3 is fed back to the input of integrator 1 (as shown by the equality of the left- and right-hand sides of equation (3)), the unknown y is found accordingly.

As an illustration of connecting a number of digital integrators, consider solution of the differential equation

$$\frac{d^2y}{dx^2} + f(x)\frac{dy}{dx} + x \cdot y = 0. \tag{8}$$

If the quantities in this equation may be represented in the form of increments, then the equation is written in the form

$$-d\left(\frac{d^2y}{dx^2}\right) = df(x)\frac{dy}{dx} + f(x)\, d\left(\frac{dy}{dx}\right) + x dy + y dx, \tag{9}$$

The solution diagram for this equation (Fig. 23) differs from the usual diagram in analog computers only in that, instead of the continuous values of integrand and independent variable, their increments are fed to the integrator inputs; and, instead of the integral itself, its increments are obtained at the integrator outputs. The increments of all terms on the right-hand side of equation (9) are summed by a summing or servo integrator at the output of which there is obtained a number of overflow pulses equal to the sum of all increments at the input. Its output is fed back to the input of integrator 1 and equation (9) is satisfied. Of extreme importance for the digital computer is the fact that not the quantities themselves but their increments in the form of a series of separate overflow pulses at the output of the integrators are transmitted over the connected lines among the integrators. This permits extremely simple and flexible "hookup" of integrators by entry of a coded program into the machine. For example, for interconnection of integrators 1 and 2 it suffices to apply an overflow pulse from the output of integrator 1 to the input of integrator 2. This is achieved by an electronic coincidence circuit which, from all the stored overflow pulses, chooses at a definite time the required pulse for the second integrator.

Solution of more complex differential equations as well as of algebraic equations and transcendental equations and transcendental equations reduces to the representation of the given equations in differential form and the determination of a solution diagram which is often a complex system with many interconnected feedback loops. Initial conditions for the integrands of the

integrators must be assigned. Functional dependences may be generated by the solution of differential equations. The methods and diagrams for obtaining the most frequently encountered functions are described below.

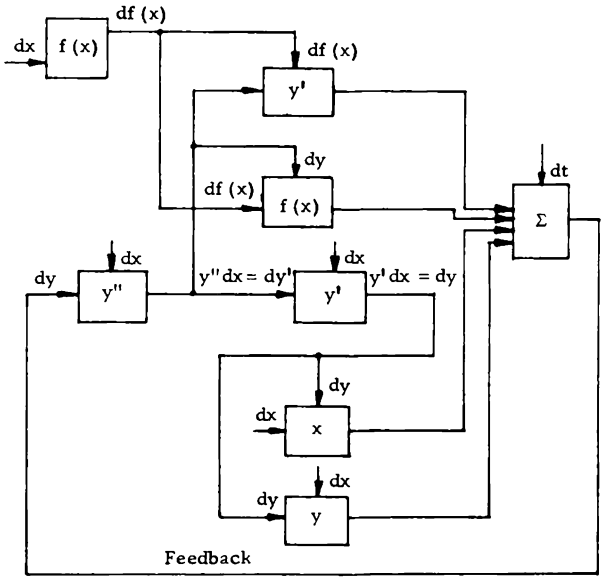


Fig. 23. Solution of the equation $y'' + y' \cdot f(x) + xy = 0$.

2. BASIC ALGEBRAIC OPERATIONS

2.1 Addition and Subtraction

Addition or subtraction of several variables is achieved by summation of their increments in a reversible counter $\sum\Delta y$, that is,

$$d\sum = dx \pm dy \pm dz \pm \ldots \tag{10}$$

Summation of six or seven increments of variables can usually be achieved at input y of the integrator. If the increment sum $d\sum$ must be fed to the y input of the next integrator, all increments are simply applied to the y input of this integrator and summed in the $\sum\Delta y$ counter; a special summing integrator is not necessary.

If the result of summation is used in any subsequent integrator as an increment of an independent variable, a summing (servo)

integrator is used (see Chapter I, Section 10), which "generates" at the output the algebraic sum of the increments at its input. The code of the largest value of a negative number (0 000...00) is written in the integrand register of the summing (servo) integrator. The initial value of the sum is written in the integrand register of that integrator in which the overflows from the adder output are accumulated. In summation of several variables their scales must be identical; the change of scale factor can be achieved by means of scaling integrators.

2.2 Multiplication

The multiplication of two variables x and y is achieved in a differential form:

$$d(xy) = xdy + ydx, \tag{11}$$

Thus, a minimum of two integrators (Fig. 24a) is required. Summation of the outputs of these two integrators may be achieved in the form of increments of the integrand at the input of the next integrator. If the increment $d(xy)$ is used as the independent variable in other integrators, summation of $xdy + ydx$ requires an additional integrator.

The finite values of increments x and y lessen the accuracy of multiplication in using formula (11) because of omission of a second-order term described below:

$$(x + \Delta x)(y + \Delta y) = xy + x\Delta y + y\Delta x + \Delta x \cdot \Delta y$$

or

$$\begin{aligned} \Delta(xy) &= x\Delta y + y\Delta x + \Delta y \Delta x = \\ &= x\Delta y + (y + \Delta y)\Delta x = x\Delta y + y^*\Delta x, \end{aligned} \tag{12}$$

where x is the previous (old) value and y^* is the new value of the multiplied numbers. Formula (12) is more accurate and is used in the parallel integrating computer, whereas formula (11) is used in the serial integrating computer (see Chapter IX, Sec. 4).

If it is necessary to multiply the product of U and V by the increment of a third variable dx (i.e., to obtain the product $UVdx$), only two integrators are needed as shown in Fig. 24b. Generation of the product $UVWdx$ requires a total of three integrators.

Multiplication of three independent variables UVW is obtained by using the formula

$$d(UVW) = (UV)dW + Wd(UV) \tag{13}$$

As is seen (Fig. 24c), six integrators, including two servo integrators, are required. The servo integrators perform summation

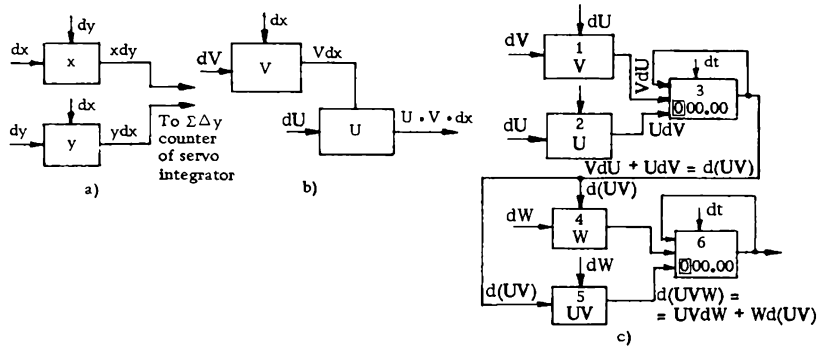


Fig. 24. Multiplication:
(a) $d(xy) = x\,dy + y\,dx$; (b) generation of $UV\,dx$; (c) generation of UVW.

of the two terms applied to their inputs. The output of integrator 6 is the increments $d(UVW)$, which are summed in the counter of another integrator delivering the product UVW under the assigned initial conditions.

2.3 Division

The division operation $W = \frac{U}{V}$ or $WV = U$ is performed by means of the multiplication operation according to the formula

$$V\,dW + W\,dV - dU = 0 \tag{14}$$

or, inserting $W = \frac{U}{V}$, we obtain

$$V\,d\left(\frac{U}{V}\right) + \frac{U}{V}\,dV - dU = 0. \tag{15}$$

This equation is solved (Fig. 25) by means of three integrators, integrator 3 being a servo integrator.

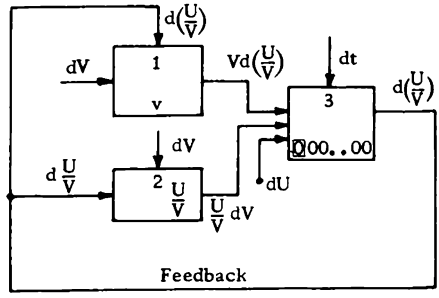


Fig. 25. Generation of quotient $\frac{U}{V}$.

3. GENERATION OF ALGEBRAIC, TRIGONOMETRIC AND OTHER FUNCTIONS. TRANSFORMATION OF COORDINATES.

Various functions (including exponential, trigonometric, hyperbolic, elliptical, complex, integral sine and cosine functions) can be generated by means of digital integrators in solving appropriate differential equations.

Below are the diagrams for generation of the functions most frequently encountered in the solution of various problems. A more detailed discussion is given in [1].

3.1 Extraction of the square root $\sqrt{U}$ (Fig. 26):

$$U = (\sqrt{U})^2 \tag{16}$$

or

$$2\sqrt{U}\,d(\sqrt{U}) - dU = 0.$$

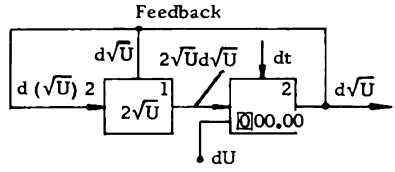


Fig. 26. Generation of function $y = \sqrt{U}$.

3.2 Squaring (Fig. 27):

$$\begin{aligned} y &= x^2; \\ dy &= 2x \cdot dx. \end{aligned} \tag{17}$$

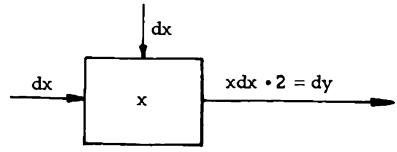


Fig. 27. Generation of function $y = x^2$.

3.3 Exponential function (Fig. 28):

$$y = e^{\pm x}$$

$$y' = \pm y$$

or

$$dy = \pm y dx. \tag{18}$$

The initial value $y_0 = e^{\pm x_0}$.

3.4 Inverse function (Fig. 29):

$$y = \frac{1}{x};$$

$$dy = -\frac{1}{x^2} dx = -y^2 dx$$

or

$$dy = -y(ydx). \tag{19}$$

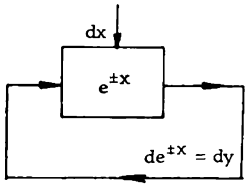


Fig. 28. Generation of function $y = e^{\pm x}$.

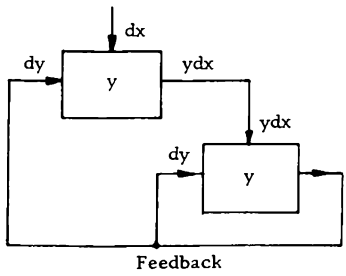


Fig. 29. Generation of an inverse function.

3.5 Inverse function and the square of an inverse function (Fig. 30):

$$y = \frac{1}{x} \text{ and } z = \frac{1}{x^2};$$

$$dy = -y^2 dx. \tag{20}$$

3.6 Inverse and logarithmic functions $\frac{1}{x}$, ln x (Fig. 31):

$$d \ln x = \frac{1}{x} dx;$$

$$d\left(\frac{1}{x}\right) = -\frac{1}{x^2} dx = -\frac{1}{x}(d \ln x). \tag{21}$$

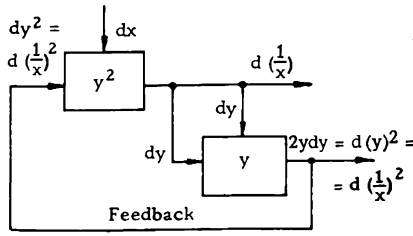


Fig. 30. Generation of the inverse function and square of an inverse function.

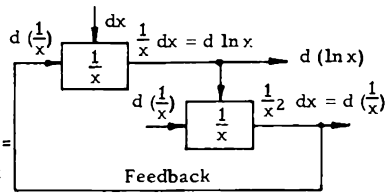


Fig. 31. Generation of an inverse function and a logarithmic function.

3.7 Probability function (Fig. 32) (probability integral):

$$y = \frac{2}{\sqrt{\pi}} \int_0^x e^{-x^2}\, dx;$$

$$dy = \frac{2}{\sqrt{\pi}} e^{-x^2}\, dx$$

or

$$y' = \frac{2}{\sqrt{\pi}} e^{-x^2};$$

$$y'' = -\frac{4}{\sqrt{\pi}} x \cdot e^{-x^2}$$

or

$$y'' = -2x \cdot y'$$

or

$$dy' = -2x dy. \tag{22}$$

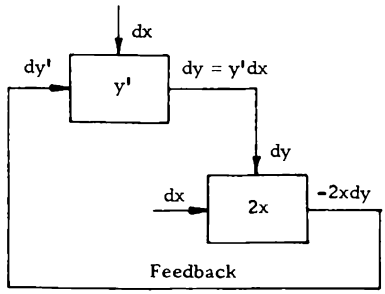


Fig. 32. Generation of probability function.

3.8 Generation of the function

$$dz = -\frac{dU}{1+V},$$

where U and V are the independent variables (Fig. 33);

$$dz(1+V) = -dU$$

or

$$dz = -V\,dz - dU. \tag{23}$$

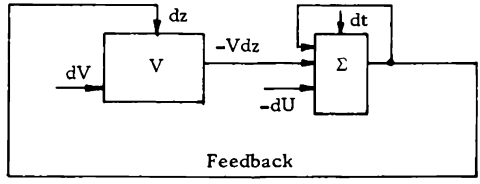


Fig. 33. Generation of the function $dz = \frac{dU}{1+U}$.

3.9 Generation of the function

$$dz = -\frac{dx}{y}$$

(Fig. 34) may be performed as:

$$(y-1)\frac{dx}{y} - dx = -\frac{dx}{y}. \tag{24}$$

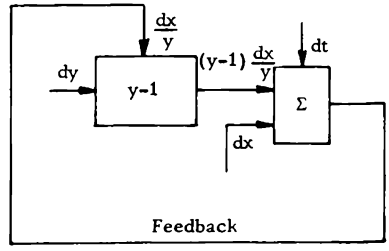


Fig. 34. Generation of the function $dz = -\frac{dx}{y}$.

3.10 Power function (Fig. 35a and b):

$$y = x^4;$$

$$x\,dx = \frac{1}{2}\,d(x^2);$$

$$x^2\,dx = \frac{1}{3}\,d(x^3);$$

$$x^3\,dx = \frac{1}{4}\,d(x^4)\,. \tag{25}$$

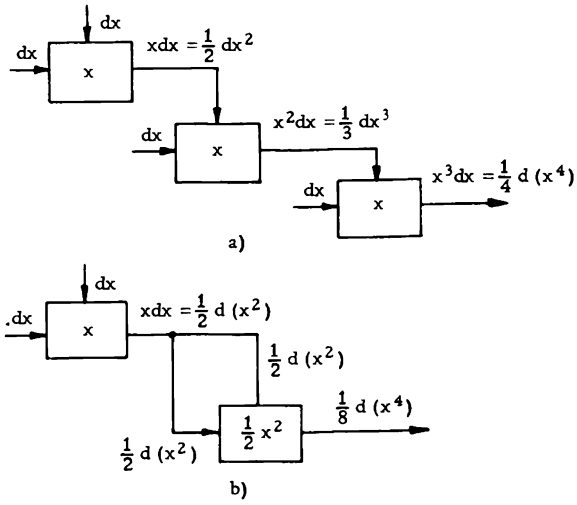


Fig. 35. Generation of a power function.

3.11 Trigonometric functions (Fig. 36):

$$y = \sin x;$$

$$y' = \cos x;$$

$$y'' = -\sin x$$

or

$$y'' = -y. \tag{26}$$

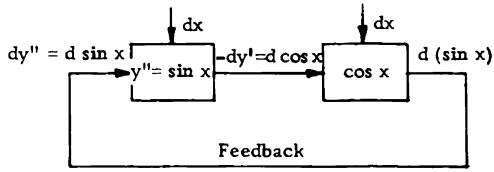


Fig. 36. Generation of function sin x and cos x.

3.12 Generation of tangent and secant (Fig. 37):

$$d\ (\tan x) = (\sec x)^2 dx;$$

$$d\ (\sec x) = \sec x \cdot \tan x \cdot dx. \tag{27}$$

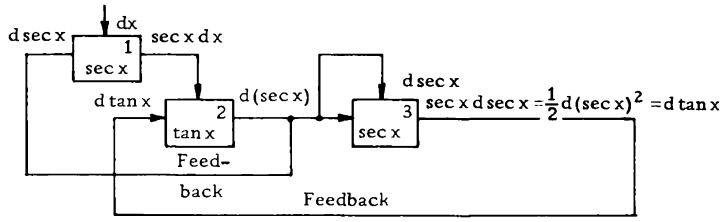


Fig. 37. Generation of tangent and secant functions.

3.13 Generation of tangent, tangent squared and secant squared (Fig. 38):

$$d\ (\tan x) = (\sec x)^2 dx; \tag{28}$$

$$\sec^2 x = 1 + \tan^2 x; \tag{29}$$

$$d\ (\tan x) = (1 + \tan^2 x)\, dx. \tag{30}$$

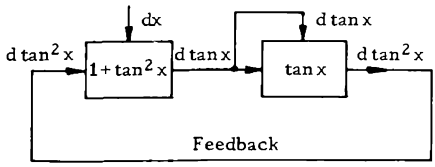


Fig. 38. Generation of tan and $\tan^2$ functions.

3.14 Generation of cotangent and cosecant (Fig. 39):

$$d\ (\cot x) = -(\csc x)^2 dx \tag{31}$$

$$d\ (\csc x) = -(\csc x)\ (\cot x)\ dx \tag{32}$$

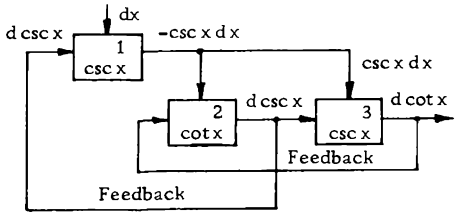


Fig. 39. Generation of cotan and cosec functions.

3.15 Generation of cotan, cotan^2 and cosec^2 (Fig. 40):

$$d \cot x = -(\csc x)^2 dx; \tag{33}$$

$$\csc^2 x = 1 + \cot^2 x; \tag{34}$$

$$d(\cot x) = (1 + \cot^2 x)\, dx. \tag{35}$$

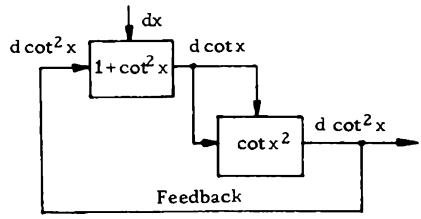


Fig. 40. Generation of the cot^2 function.

3.16 Generation of arc-sine, $x = \text{arc sin } y$ (Fig. 41):

$$y = \sin x;$$

$$dy = \cos x dx;$$

$$dx = \frac{d(\sin x)}{\cos x} = \sec x \cdot d(\sin x), \tag{36}$$

where y is unknown and x is given.

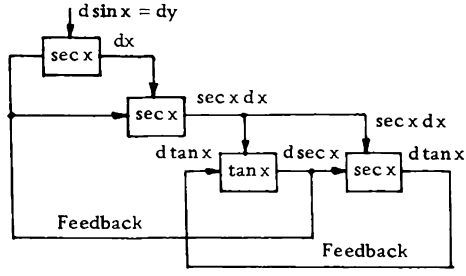


Fig. 41. Generation of the inverse sine function.

Another method of generating $x = \text{arc sin } y$ is by means of a servo integrator (Fig. 42):

$$\left.\begin{aligned} x &= \text{arc sin } y, \\ y &= \sin x, \\ dy &= d(\sin x). \end{aligned}\right\} \tag{37}$$

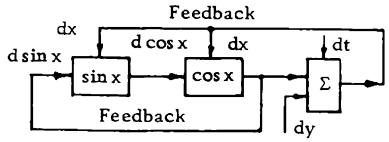


Fig. 42. Generation of the inverse sine function by means of a servo integrator.

3.17 Generation of arc-cosine, $x = \arccos y$ (Fig. 43):

$$y = \cos x;$$

$$dy = d(\cos x) = -\sin x\, dx;$$

$$dx = -\frac{d(\cos) x}{\sin x};$$

$$dx = -\csc x\, dy, \tag{38}$$

where y is given and x is unknown.

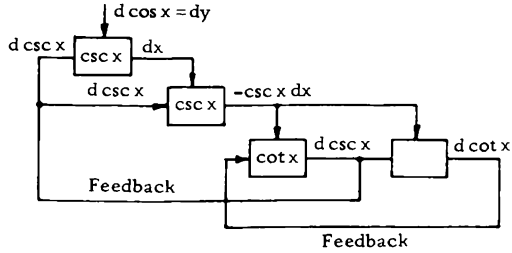


Fig. 43. Generation of the inverse cosine function.

3.18 Generation of arc-tangent, $x = \arctan y$ (Fig. 44):

$$y = \tan x;$$

$$dy = \sec x\, dx;$$

$$dx = \cos x\, dy, \tag{39}$$

where y is given and x is unknown.

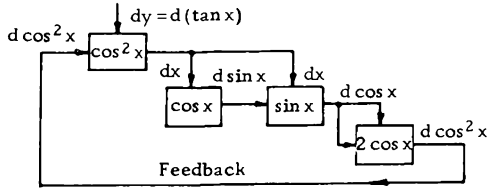


Fig. 44. Generation of the inverse tangent function.

3.19 Generation of arc-secant, $x = \text{arc sec } y$ (Fig. 45):

$$y = \sec x;$$

$$dy = \sec x \cdot \tan x \cdot dx;$$

$$dx = \cos x \cdot \cot x \cdot dy, \tag{40}$$

where y is given and x is unknown.

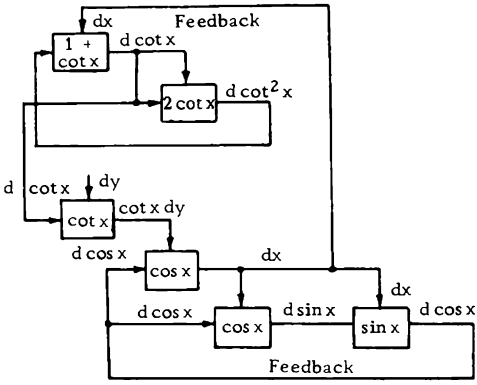


Fig. 45. Generation of inverse secant function.

3.20 Generation of arc-cosecant, $x = \text{arc csc } y$ (Fig. 46):

$$y = \csc x;$$

$$dy = -\csc x \cdot \cot x \cdot dx;$$

$$dx = \sin x \cdot \tan x \cdot dy;$$

$$d(\tan x) = (1 + \tan^2 x)\, dx, \tag{41}$$

where y is given and x is unknown.

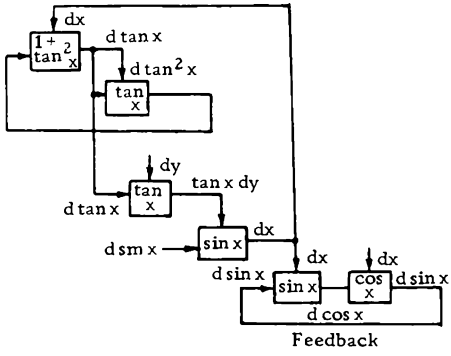


Fig. 46. Generation of inverse cosecant function.

3.21 Hyperbolic functions (Fig. 47):

$$y = \sinh x = \frac{e^x - e^{-x}}{2}; \tag{42}$$

$$\cosh x = \frac{e^x + e^{-x}}{2}; \tag{43}$$

$$dy = d(\sinh x) = \cosh x \cdot dx; \tag{44}$$

$$dy' = d(\cosh x) = \sinh x \cdot dx. \tag{45}$$

The differential equation

$$y'' = +y. \tag{46}$$

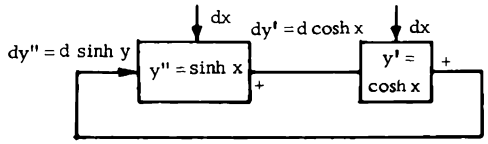


Fig. 47. Generation of the hyperbolic sine and cosine functions.

3.22 Integral sine function (Fig. 48):

$$y = \int \frac{\sin x}{x} dx; \tag{47}$$

$$dy = \frac{\sin x}{x} dx = \frac{d(\cos x)}{x}; \tag{48}$$

$$x dy = d(\cos x). \tag{49}$$

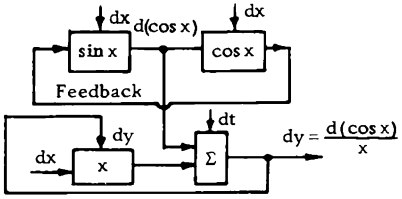


Fig. 48. Generation of the integral sine function.

3.23 Bessel functions. The Bessel equation:

$$\frac{d^2y}{dx^2} + \frac{dy}{x dx} + \left(1 - \frac{n^2}{x^2}\right) y = 0; \tag{50}$$

$$dy' + \frac{dy}{x} + ydx - yn^2\frac{dx}{x^2} = 0 \tag{51}$$

or

$$dy' = -\frac{dy}{x} - ydx - n^2yd\left(\frac{1}{x}\right). \tag{52}$$

Solving this equation, we find the Bessel function (Fig. 49).

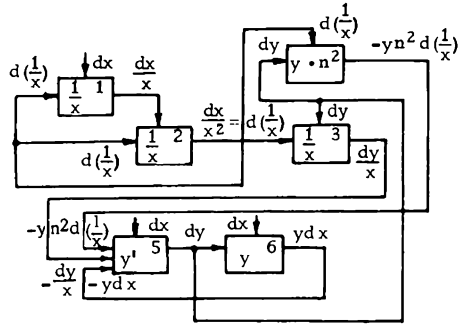


Fig. 49. Solution of the Bessel equation.

3.24 Elliptic functions (Jacobian elliptic function) (Fig. 50):

$$U = \int_0^{\varphi} \frac{d\varphi}{\sqrt{1-k^2\sin^2\varphi}} = \int_0^{x=\sin\varphi} \frac{dx}{(1-x^2)\sqrt{1-k^2x^2}}; \tag{53}$$

$$E = \int_0^{\varphi} \sqrt{1-k^2\sin^2\varphi}\,d\varphi = \int_0^{x=\sin\varphi} \sqrt{\frac{1-k^2x^2}{1-x^2}}\,dx. \tag{54}$$

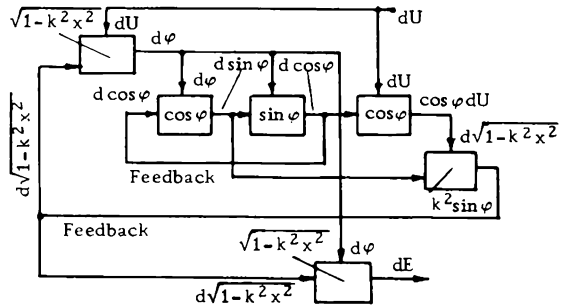


Fig. 50. Generation of Jacobian elliptic functions.

3.25 Complex functions (Fig. 51)

Given the complex function

$$x = \varrho e^{i\varphi} = e^{\Phi + i\varphi}, \tag{55}$$

where $\varrho = e^{\Phi}$. Find the real and imaginary components of the complex quantity (vector)

$$x_r + ix_i = e^{\Phi + i\varphi}. \tag{56}$$

From equation (56) we find

$$\begin{aligned} dx_r + idx_i &= (e^{\Phi + i\varphi})(d\Phi + id\varphi) \\ dx_r + idx_i &= (x_r + idx_i)(d\Phi + id\varphi) = \\ &= x_r d\Phi + idx_i d\Phi + x_r id\varphi + i^2 dx_i d\varphi. \end{aligned} \tag{57}$$

Hence, equating the real and imaginary parts of the equation, we obtain

$$\left.\begin{aligned} dx_r &= x_r d\Phi - x_i d\varphi, \\ dx_i &= x_i d\Phi + x_r d\varphi. \end{aligned}\right\} \tag{58}$$

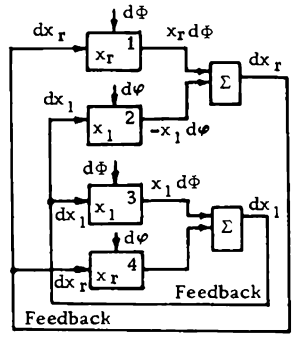


Fig. 51. Generation of a complex function.

3.26 Polynominal of *n*th degree

As an example, consider a third degree polynomial (Fig. 52):

$$y = ax^3 + bx^2 + cx + d; \tag{59}$$

$$y' = 3ax^2 + 2bx + c; \tag{60}$$

$$dy = (3ax^2 + 2bx + c)\,dx; \tag{61}$$

$$y'' = 6ax + 2b; \tag{62}$$

$$dy' = (6ax + 2b)\,dx; \tag{63}$$

$$y''' = 6a; \tag{64}$$

$$dy'' = 6a\,dx. \tag{65}$$

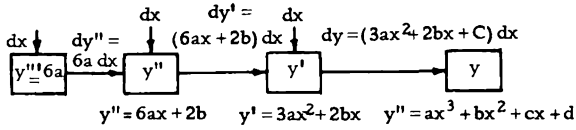


Fig. 52. Generation of a third degree polynomial.

3.27 Solution of right triangle and coordinate transformation (Fig. 53a, b and c):

$$r^2 = x^2 + y^2; \tag{66}$$

$$x = r \cdot \cos\alpha; \tag{67}$$

$$y = r \cdot \sin\alpha; \tag{68}$$

$$\tan\alpha = \frac{y}{x} = \frac{\sin x}{\cos x}. \tag{69}$$

a) Determining α from given r and x:

$$d\alpha = \frac{\cos\alpha\,dr - dx}{r\sin\alpha}. \tag{70}$$

b) Determining α from given r and y:

$$d\alpha = \frac{dy - \sin\alpha\,dr}{r\sin\alpha}. \tag{71}$$

c) Determining α from given x and y:

$$d\alpha = \frac{\cos\alpha\,dy - \sin\alpha\,dx}{x\cos\alpha + y\sin\alpha}. \tag{72}$$

d) Determining r from given x and y:

$$dr = \frac{x\,dx + y\,dy}{r}, \tag{73}$$

e) Determining r from given α and x:

$$dr = \frac{dx - r\,d(\cos\alpha)}{\cos\alpha}. \tag{74}$$

f) Determining r from given α and y:

$$dr = \frac{dy - rd(\sin\alpha)}{\sin\alpha}, \tag{75}$$

g) Determining x from given r and y:

$$dx = \frac{rdr - ydy}{x}, \tag{76}$$

h) Determining x from given r and α:

$$dx = rd(\cos\alpha) + \cos\alpha\, dr. \tag{77}$$

i) Determining x from given α and y:

$$dx = \frac{yd(\cos\alpha) + \cos\alpha\, dy - xd(\sin\alpha)}{\sin\alpha}, \tag{78}$$

j) Determining y from given r and α:

$$dy = rd(\sin\alpha) + \sin\alpha\, dr. \tag{79}$$

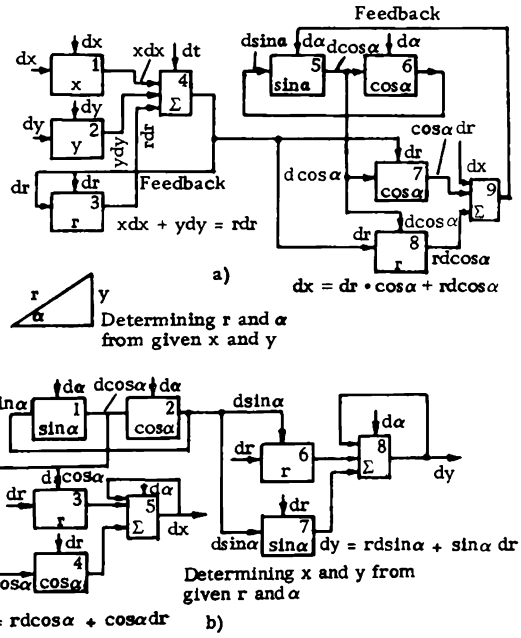


Fig. 53. Coordinate transformation.

3.28 Transformation of spherical (polar) coordinates to Cartesian.

In the transformation of polar coordinates in space it is often necessary to determine not only the Cartesian coordinates x and

y but also the rate of change in the coordinates, i.e., the derivatives $\frac{dx}{dt}$ and $\frac{dy}{dt}$ (Fig. 54).

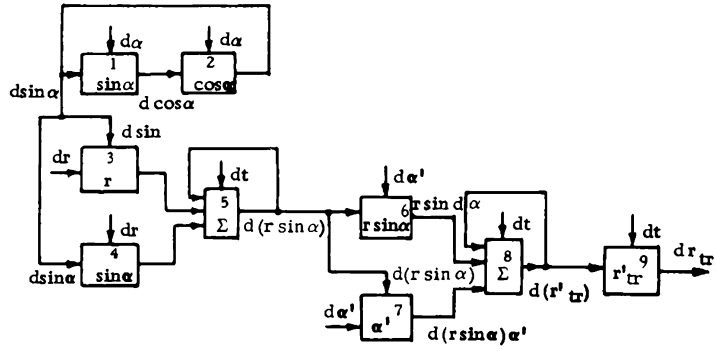


Fig. 54. Determining the projection of a space vector onto a plane.

In order to avoid the differentiation operation let us use the formulas listed below.

Initial formula: Differential form:

$$x = r_{tr}\cos\beta \qquad (80) \qquad r'_{tr} = r\sin\alpha\cdot\alpha' + \cos\alpha\cdot r' \qquad (81)$$

determination of r requires five integrators (Fig. 54);

$$y = r\cdot\sin\alpha \qquad (82) \qquad \frac{dx}{dt} = z\beta' + \cos\alpha\, r'_{tr} \qquad (83)$$

determination of $\frac{dx}{dt}$ and dx requires five integrators;

$$z = r_{tr}\sin\beta \qquad (84) \qquad \frac{dy}{dt} = r_{tr}\alpha' + \sin\alpha\cdot r' \qquad (85)$$

determination of $\frac{dy}{dt}$ and dy requires five integrators;

$$r_{tr} = r\cos\alpha \qquad (86) \qquad \frac{dz}{dt} = r_{tr}\sin\beta - x\beta' \qquad (87)$$

determination of $\frac{dz}{dt}$ and dz requires five integrators.

In addition, for generation of

$$\sin\alpha,\quad \cos\alpha \text{ and } \sin\beta,\quad \cos\beta$$

four integrators are required.

Solution of equations (80) to (88) requires 26 integrators (Figs. 55 and 56).

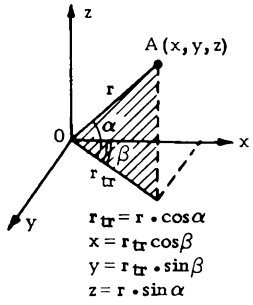


Fig. 55. Transformation of polar coordinates in space to Cartesian coordinates.

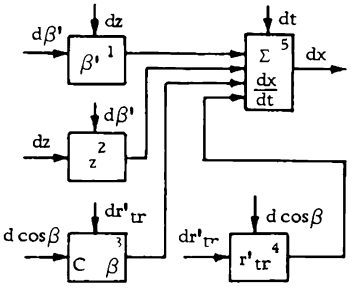


Fig. 56. Transformation of Cartesian coordinates in space to polar coordinates (solution of equation 83).

3.29 Transformation of Cartesian coordinates to polar coordinates

Given x, y and z, determine r, β and α.

Initial formula: Differential form:

$$r^2 = x^2 + y^2 + z^2 \qquad (89) \qquad rdr = xdx + ydy + zdz \qquad (90)$$

determination of dr and r requires four integrators (Fig. 57a);

$$r_{tr}^2 = x^2 + y^2; \qquad (91) \qquad r_{tr} dr_{tr} = xdx + ydy, \qquad (92)$$

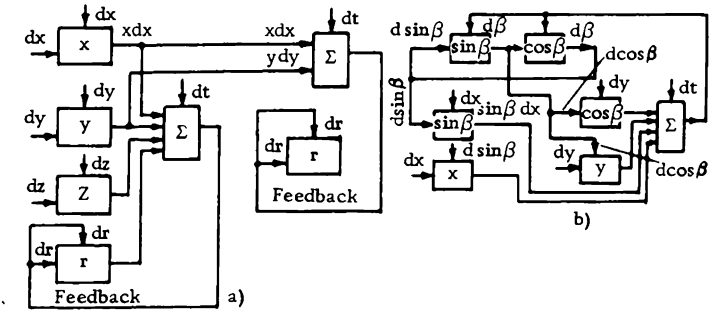


Fig. 57. Diagrams for solution of equations:

determination of dr_{tr} and r requires two integrators;

$$x = r_{tr}\cos\beta \qquad (93) \qquad dx = r_{tr}d(\cos\beta) + \cos\beta\, dr_{tr} \qquad (94)$$

determination of $d\beta$ and β requires five integrators (Fig. 57b);

$$z = r\cdot\sin\alpha \qquad (95) \qquad dz = r\cdot d(\sin\alpha) + \sin\alpha\, dr \qquad (96)$$

determination of $d\alpha$ and α requires five integrators. Solution of all these equations requires 16 integrators.

4. SOLUTION OF LINEAR FIRST-ORDER ALGEBRAIC EQUATIONS

In order to solve a system of linear first-order algebraic equations, they must be represented in a differential form. Let the system of equations be:

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + A_1 = 0,\\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + A_2 = 0,\\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + A_3 = 0. \end{aligned}\right\} \qquad (97)$$

Multiply both sides of these equations by dt:

$$\left.\begin{aligned} a_{11}x_1dt + a_{12}x_2dt + a_{13}x_3dt + A_1dt = 0,\\ a_{21}x_1dt + a_{22}x_2dt + a_{23}x_3dt + A_2dt = 0,\\ a_{31}x_1dt + a_{32}x_2dt + a_{33}x_3dt + A_3dt = 0. \end{aligned}\right\} \qquad (98)$$

Solution of these equations is achieved by means of an arrangement (Fig. 58) containing $(n+1)^2$ integrators for n equations with $n+1$ terms in each equation. It is assumed that increments at the inputs dx_1, dx_2 and dx_3 of the integrators are summed in the increment counter for the integrand without the use of summing integrators. Approximate values may be assigned as the initial values of x_1, x_2 and x_3. These approximate values may be preliminarily obtained by solution of the equations in a low-accuracy machine using a small number of digits and a small number of iterations. Once the approximate values of the unknowns have been determined, they are inserted as the initial data; the solution is repeated using the total number of digits in a high-accuracy machine. In both cases, negative feedback must be used in solution of the equations. In short, this method of solution gives a rapid determination of accurate values of the unknowns if their approximate values are first determined.

The second method of solution of these equations consists in representing them in the form of a sum of increments of the unknowns multiplied by the corresponding coefficients, i.e.:

$$\left.\begin{aligned} a_{11}dx_1 + a_{12}dx_2 + a_{13}dx_3 = 0,\\ a_{21}dx_1 + a_{22}dx_2 + a_{23}dx_3 = 0,\\ a_{31}dx_1 + a_{32}dx_2 + a_{33}dx_3 = 0. \end{aligned}\right\} \qquad (99)$$

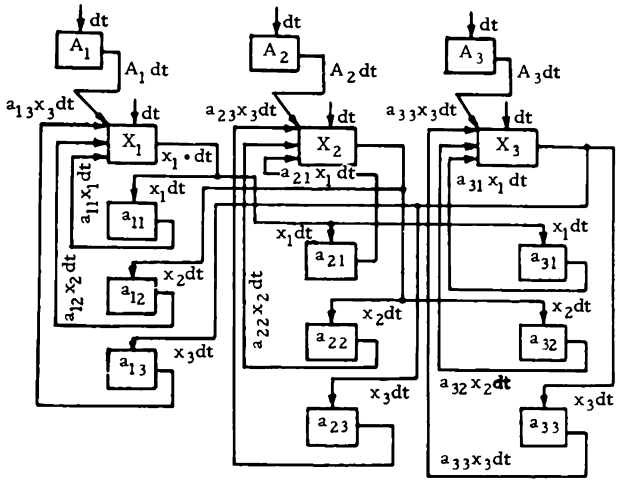


Fig. 58. Solution of a system of linear algebraic equations.

The unknowns of these equations (Fig. 59) are determined by means of servo integrators. The values of the absolute terms A_1, A_2 and A_3 are first introduced in the registers of these servo integrators. Increments dx_1, dx_2 and dx_3 are accumulated in counters x_1, x_2 and x_3 and generate the values of the unknowns. The initial values in counters x_1, x_2 and x_3 are set at zero. The diagram in Fig. 59 is similar to that for solution of a system of linear algebraic equations by means of the servo systems used in an analog computer.

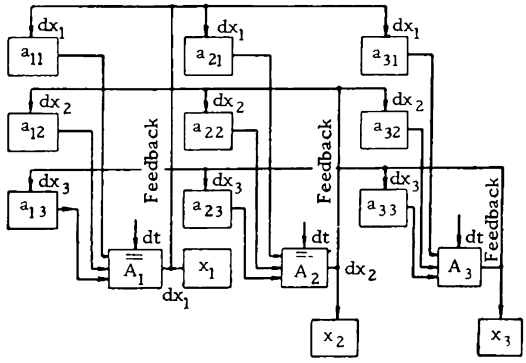


Fig. 59. Variant arrangement for solution of a system of linear algebraic equations.

A disadvantage of this method is the fact that the number of iterations required to find the accurate values of the unknowns may be large; as a result, the solution time is increased.

5. SOLUTION OF HIGHER-ORDER ALGEBRAIC EQUATIONS

Determination of the real or complex roots of an nth order equation

$$a_0x^n + a_1x^{n-1} + \ldots + a_{n-1}x + a_n = 0 \tag{100}$$

reduces to finding the vector

$$x = \beta e^{i\varphi} = \rho(\cos\varphi + i\sin\varphi) \tag{101}$$

which satisfies the given equation.

In this case the vector sum of the individual terms of the equation must be equal to zero, i.e.,

$$a_0\rho^n e^{in\varphi} + a_1\rho^{n-1}e^{i(n-1)\varphi} + \ldots + a_{n-1}e^{i\varphi} + a_n = 0. \tag{102}$$

This equation can be written into two equations whose simultaneous solution determines the roots. For example, the equation

$$a_0x^3 + a_1x^2 + a_2x + a_3 = 0 \tag{103}$$

is written into the following two equations:

$$\left.\begin{aligned} a_0\rho^3\cos 3\varphi + a_1\rho^2\cos 2\varphi + a_2\rho\cos\varphi + a_3 = 0, \\ a_0\rho^3\sin 3\varphi + a_1\rho^2\sin 2\varphi + a_2\rho\sin\varphi = 0. \end{aligned}\right\} \tag{104}$$

Multiplying both sides of these equations by dt, we obtain

$$\left.\begin{aligned} a_0\rho^3\cos 3\varphi\,dt + a_1\rho^2\cos 2\varphi\,dt + a_2\rho\cos\varphi\,dt + a_3dt = 0, \\ a_0\rho^3\sin 3\varphi\,dt + a_1\rho^2\sin 2\varphi\,dt + a_2\rho\sin\varphi\,dt = 0. \end{aligned}\right\} \tag{105}$$

These equations are solved simultaneously. The setup diagram for one of the equations is shown in Fig. 60. Summing of the increments of the individual terms of the equation may be performed directly at the integrand input of the integrators without the use of a summing integrator. The setup in Fig. 60 is used in determining ρ; a similar setup is used in determining φ, $\cos\varphi$ and $\sin\varphi$.

Let us discuss the solution of nonlinear equations given in the form

$$\left.\begin{aligned} U = f(x, y) = 0, \\ V = \varphi(x, y) = 0. \end{aligned}\right\} \tag{106}$$

By means of partial derivatives the above equations may be written in the form

$$\left.\begin{aligned} dU &= f_x dx + f_y dy, \\ dV &= \varphi_x dx + \varphi_y dy \end{aligned}\right\} \tag{107}$$

and solved by means of the setup in Fig. 61. The values of functions $U = f(x_0, y_0)$ and $V = \varphi(x_0, y_0)$ for initial values of x_0 and y_0 may be inserted into the servo integrator registers.

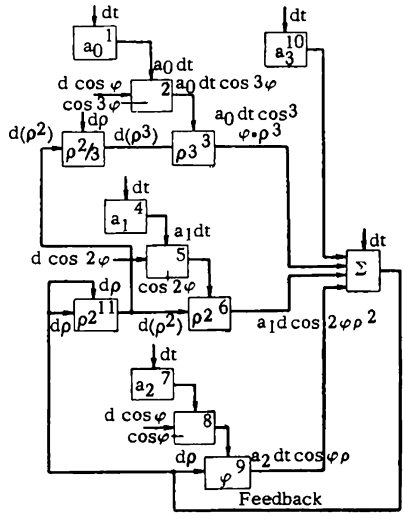


Fig. 60. Solution of higher-order algebraic equations $a_0 \rho^3 \cos 3\varphi\, dt + a_1 \rho^2 \cos \varphi\, dt + a_2 \rho \cos \varphi\, dt + a_3\, dt = 0$.

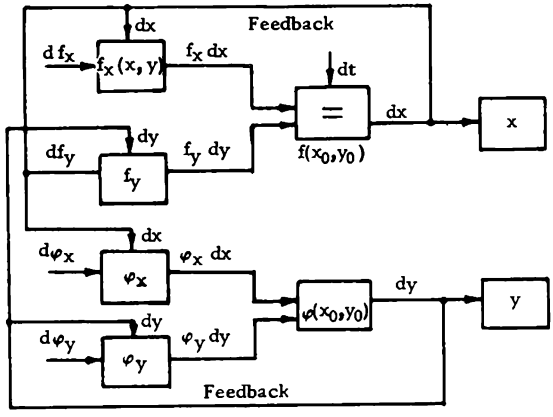


Fig. 61. Solution of nonlinear algebraic equations with functional dependences:
$df = f_x dx + f_y dy$; $d\varphi = \varphi_x dx + \varphi_y dy$.

6. SOLUTION OF DIFFERENTIAL EQUATIONS WITH GIVEN BOUNDARY (INITIAL AND FINAL) CONDITIONS

Solution of a differential equation with given boundary (initial and final) conditions reduces to the problem of finding from a family of curves which satisfy the given differential equation the curve that corresponds to the given boundary conditions.

In order to find the required curve it is necessary to perform computations for the family of curves by varying the initial conditions. This process is continued until the required curve is found. The search may be made automatically and the solution time is reduced by the use of integrators containing a register and an adder for storage of new initial data (see Chapter V, Sec. 12). As an illustration, let us solve the differential equation:

$$\frac{d^2y}{dx^2} = A \cdot x + e^y + C \tag{108}$$

or

$$d\left(\frac{dy}{dx}\right) = (Ax + e^y + C)\,dx. \tag{109}$$

with the given boundary conditions:

a) initial conditions

$$\left.\begin{array}{l}\left(\frac{dy}{dx}\right)_0 = -B \\ \text{for } x_0 = 0\end{array}\right\} \tag{110}$$

b) final conditions

$$\left.\begin{array}{l}\left(\frac{dy}{dx}\right)_\kappa = 0, \quad y_\kappa = 0 \\ \text{for } x = x_\kappa.\end{array}\right\} \tag{111}$$

The given problem is to find y_0 and x_κ for the curve which satisfies the boundary conditions and equation (108). Or it is to find the curve in the family of curves (Fig. 62) which is tangent to the x-axis at point x_κ, i.e., curve y at point x_κ has $y_\kappa = 0$ and $y_\kappa' = 0$. The initial value for the derivative of this curve is $y_0' = -B$ for $x = 0$.

The setup for the solution of equation (109) (Fig. 63) assumes that y is known. By means of integrators 1-3, the right side of equation (109), which is also $d\left(\frac{dy}{dx}\right)$, is generated. Then, by means of integrator 4, we obtain $\frac{dy}{dx}\,dx = dy$. Increment dy is fed back as the independent variable of the input to integrator 2. At the beginning of the solution, a chosen initial value* is inserted into

---

*The initial value of $\frac{dy}{dx}$ is computed from the formula:

$$\frac{dy}{dx} = \int (Ax + e^y + C)\,dx; \quad \text{for } x = 0.$$

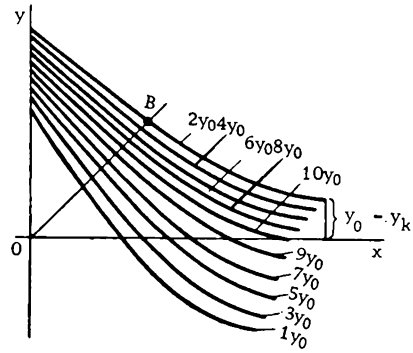


Fig. 62. Family of curves satisfying the solution of the differential equation $y'' = Ax + iy + c$.

integrators 3 and 4. The initial value y_0 is inserted into servo register 5. Since dy is applied at the input of servo integrator 5, a number equal to $y - y_0$ is obtained in its register, where y is the current value; the increment sum equal to $-(\sum dy - y_0) = y_0 - \sum dy$ is formed at its output. When the number in the register of integrator 4 becomes equal to 0 (more accurately, $\boxed{0}000...00$), at the integrator output appears a quantity equal to $y_0 - \sum dy$, where at this moment the current value of y is equal to the final value $y_к$ for a corresponding value $x_к$. Thus, the sum of the increments of the output of integrator 5 is equal to the difference between the initial value y_0 and the final value $y_к$, that is, $\Delta y_1 = y_0 - y_к$ (up to and including the point where $y' = 0$). By means of integrator 6, this difference is multiplied by a certain coefficient $m < 1$ for smoothing the fluctuations in the output quantity of integrator 5 about the "zero" position of the servo integrator. The value of $\Delta y_0 = m(y_0 - y_к)$ obtained at the output of integrator 6 is inserted continuously into the initial value register of integrator 5 and is continuously applied to the independent variable input of integrator 7. The output of integrator 7 is a continuous generation of increments Δe^{y_0} which are accumulated in the initial condition register of integrator 2 and summed with the initial value e^{y_0}. The value $-e^{y_0}$ is chosen as the initial value of the integrand in integrator 2 for a certain initial value of y_0. Since y decreases with a change in x from x_0 to $x_к$, e^y also decreases. After each "curve run" on the machine, new initial values for y_0 and e^{y_0}, obtained as the result of accumulation of increments Δy_0 and Δe^{y_0} in the initial value register, are inserted into integrators 2 and 5. Thus, the initial value registers of integrators 2 and 5 serve the function of continuous accumulation of increments Δy_0 and Δe^y in the process

of problem solution (i.e., in the process of the "run" of each curve of the family of curves). Insertion of the accumulated new initial values for the next solution is achieved discretely upon transition from one curve to another. Computations are performed until the given boundary conditions of the problem are satisfied. In this case: (a) the value of the difference $\sum dy - y_0$ in the register of servo integrator 5 must be equal to zero and, consequently, $y_\kappa = 0$ for $x = x_\kappa$; (b) the value of the difference $\sum dy' - B$ in register 8 must be equal to zero and, consequently, $y_0' = -B$; $y_\kappa' = 0$ for $x = x_\kappa$.

Upon conclusion of computation the machine is automatically stopped since only "zero" will reach registers 5 and 8.

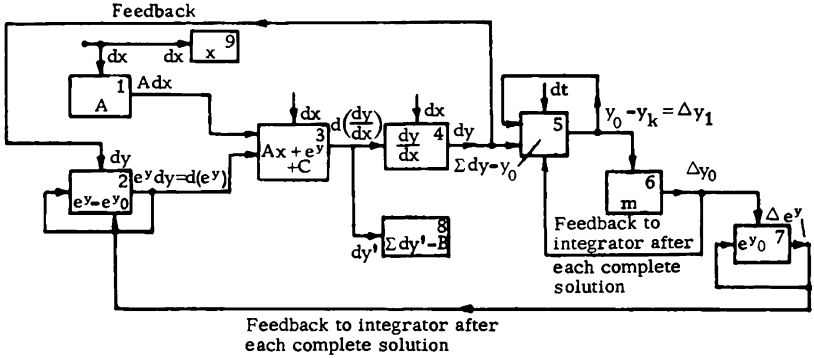


Fig. 63. Setup diagram for solution of differential equation $y'' = Ax + e^y + c$ with given boundary conditions.

7. THE DIFFERENTIATION PROCESS

Differentiation is an operation which is considerably less accurate than integration because the difference $\Delta f = f_2 - f_1$ of two extremely close values of the function $f(x)$ may have an error of the order of several percent, whereas the absolute values of the function are computed with an error several orders of magnitude smaller. In differentiation, a derivative on the curve is often obtained which differs considerably from the actual curve due to "excursion" of isolated points from a smooth curve, fluctuations of the function about a certain mean value, etc.

Hence in differentiation of a given function (for example, in the form of an experimental curve) it is first necessary to "smooth" the curve, average the observed values, etc. The curve so obtained must be sufficiently smooth. Differentiation is necessary, for example, in the determination of the rate of motion from given points of a trajectory, i.e., determination of velocity from the path or determination of acceleration from the velocity.

If we are given the increments df of function $f(x)$, the derivative is

$$f'(x) = \frac{df}{dx} \tag{112}$$

or

$$f'(x)\,dx = df$$

and

$$\int f'(x)\,dx = f(x). \tag{113}$$

Thus, the integral curve of a derivative obtained by differentiation must coincide with the function $f(x)$. The simplest setup for the differentiation operation (Fig. 64a) is one using a servo integrator (see Chapter I, Sec. 10). Servo integrator 2 compares the given value of increment df with the computed value $df = f'(x)$ in integrator 1. Since the servo integrator gives the increment at the output for each pulse dt, the accumulation of pulses in the register of integrator 1 is considerably more rapid than the change in independent variable x, and the generation of $f'(x)$ is accelerated. However, the frequency of dx must be greater than df, which is achieved by proper selection of the scale for the curve $y = f(x)$. In the actual differentiation circuit, as a rule, oscillations may occur in control of the quantity $f'(x)$, since the difference $\overline{df} - df$

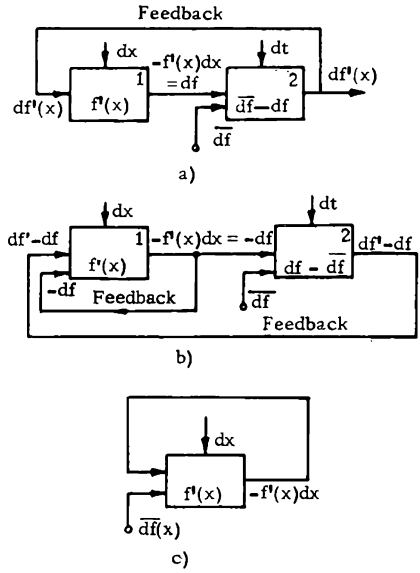


Fig. 64. Differentiation setup: a—use of a servo integrator; b—feedback in first integrator; c—differentiation with the use of one integrator.

in the register of the servo integrator may change sign. In order to eliminate oscillations about the true value of $f'(x)$, it is advisable to include integrator 1 (Fig. 63b) in a feedback loop, thereby damping these oscillations. The output of servo integrator 2 is then fed back to the input of integrator 1 in the form of increments which are proportional to the sum of increments of the function and its first derivative (i.e., to $df + df'$). This approach is similar to that in which the individual stages of a multi-stage amplifier are included in feedback loops in order to damp its oscillations.

For approximate differentiation, we may use one integrator instead of two integrators (Fig. 64c). Comparison of $\overline{df}(x)$ and $f'(x)\,dx$ is here achieved directly in counter $\sum \Delta y$ at the integrator input where the positive increments $\overline{df}(x)$ are compensated by negative increments of $f'(x)\,dx$ from the integrator output and vice versa. The diagram in Fig. 64c for this integrator is similar to that for the generation of an exponential function (see Sec. 3) in which the signal is exponentially damped.

More complex differentiation setups include a correction circuit for the error occuring during differentiation. The correcting signal generated by this circuit is proportional to the error integral and is similar to the signal obtained in error integration in ordinary control systems (to decrease dynamic error in steady-state motion).

An example of such a differentiation circuit with a correction proportional to the error integral is shown in Fig. 65. Here integrator 1 serves for approximate differentiation. Servo integrator 2 achieves generation and inversion of the error signal $\overline{df} - f'(x)\,dx = -\delta$.

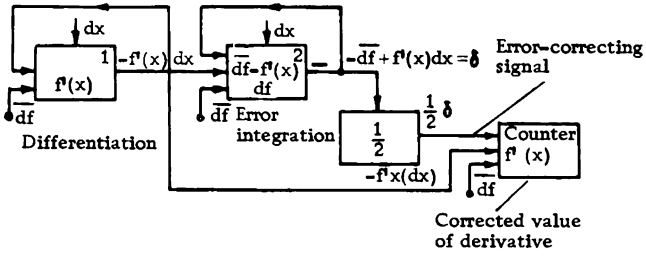


Fig. 65. Correction of error in differentiation.

In the next integrator the value of error δ is multiplied by a constant coefficient equal to $\frac{1}{2}$. After multiplication, the error signal $\frac{\delta}{2}$ is applied to a counter in which it is summed with the increment $\overline{df}'(x) = \overline{df} - f'(x)\,dx$ (i.e., with the increment of the desired derivative). The equation $df'(x) = \overline{df} - f'(x)\,dx$ follows from the fact that the two quantities in its righthand side generate the derivative in integrator 1 during integration.

Thus, the increments of the desired derivative and half of the error signal are summed in the counter. The coefficient $\frac{1}{2}$ for the error signal is taken for averaging purpose. The reading of counter 4 gives the desired value of the derivative with correction.

If function $f(x)$ is given in the form of an experimentally obtained graph inserted into a graphical function unit from which increment $df(x)$ is read out, determination of the derivative by differentiation may be achieved approximately by means of the setup in Fig. 66. In DDA applications it is important to increase the frequency of increment pulses df obtained from the external function unit. In the setup of Fig. 66 integrators 1 and 2 perform the multiplication of increments by a certain coefficient K greater than 1. These increments arrive at integrator 3 by means of which there is also performed the differentiation operation. In addition to the feedback signal — $f'(x)\,dx$ which is compared with increment $K \cdot df$, two correcting signals are applied to the input of integrator 3: a signal for correction of differentiation error and a signal for correction of drift arising with small deviations of the computed value of function increment from the given value.

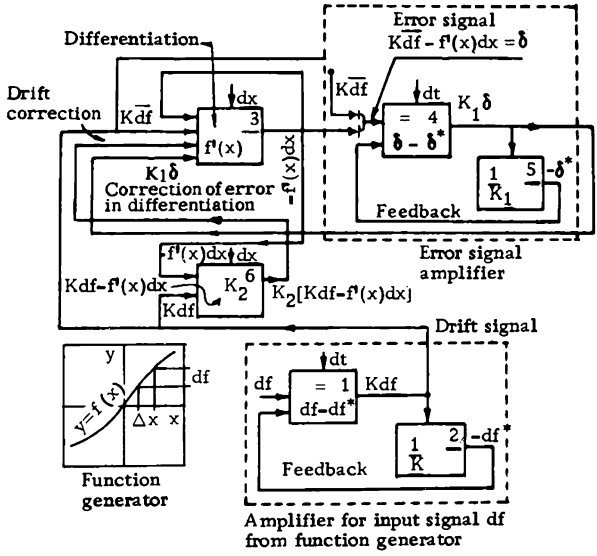


Fig. 66. Differentiation of graphically given function.

The error signal $K \cdot \overline{df} - f'(x)\,dx$ is multiplied by means of integrators by the coefficient K_1, where K_1 is greater than one (usually $K_1 = 2$ or greater, depending on the maximum value of the second derivative). The resulting value of $K_1 \cdot \delta$ is fed back to the input of integrator 3 as an error correcting signal, which is here integrated and yields the error integral.

The drift signal is the signal of error in differentiation decreased many times by multiplication with a coefficient K_2 much smaller than one (e.g., $K_2 = 0.01$) in integrator 6. The drift signal corrects the relatively small errors arising due to drift of the computed integral values about the given values.

Before differentiation of the given curve $y = f(x)$, it must first be smoothed by means of filters or other methods averaging the given data. The computation of velocity and acceleration from a given position (path) of a moving body is of great importance to engineering. However, differentiation should be avoided if there are other more accurate means of obtaining the derivatives.

8. DETERMINATION OF EXTREME VALUES

In automatic control problems it is often necessary to determine the optimum characteristics of the controlled process. This determination can be reduced to finding the maximum or minimum values of functions given in the form of equations which may be solved by a computer, or of functions given by experimental plots and tables; the unknown functional dependence may be determined by digital simulation in a DDA in conjunction with a real process.

Let us consider a process which is represented by a function z of two independent variables x and y, i.e.,

$$z = f(x, y). \tag{114}$$

As is known, the extrema of function z correspond geometrically to the "peaks" and "valleys" of a surface representing this function. The total differential df of the function is,

$$df(x, y) = \frac{df}{dx} + \frac{df}{dy}dy = f'_x dx + f'_y dy. \tag{115}$$

In order to find the values of x and y at which the function $f(x, y)$ may attain an extremum, it is necessary to equate its derivatives to zero and solve the two resulting equations with two unknowns, namely:

$$\left.\begin{aligned} f'_x(x, y) &= 0, \\ f'_y(x, y) &= 0. \end{aligned}\right\} \tag{116}$$

The real roots of these two equations are the values of x and y where an extremum occurs. The setup for solving these two equations in a DDA (Fig. 67) consists of two function generators 1 and 2, which generate f_x' and f_y' from the given x and y or their increments dx and dy (with the given initial conditions). For increments dx and dy as the inputs, the function generator gives at the output df_x' and df_y' as the increments of the partial derivatives;

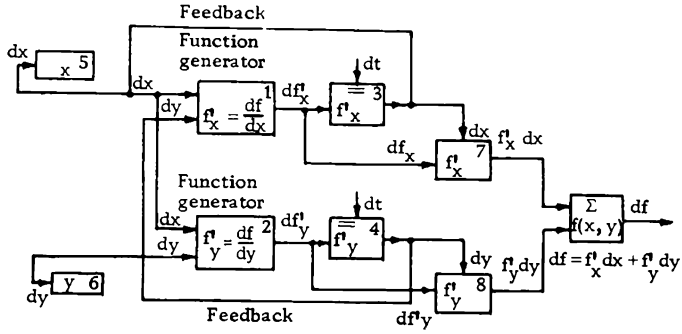


Fig. 67. Determination of extreme points for a function with two variables $z = f(x, y)$.

the partial derivatives, like the function itself, are assumed to be known. If these functions are unknown, they may be computed by means of known algorithms or determined by analog computation. Increments df_x' and df_y' are applied to the input of servo integrators 3 and 4. The increments at the output of these integrators are fed back to the inputs of function generators 1 and 2 as the increments dx and dy. When f_x' is positive, increments dx at the output of integrator 3 must increase x and vice-versa; a similar situation applies to integrator 4. When the system of two equations (116) with two unknowns x and y is solved, the increments at the output of the servo integrators are equal to zero (for $f_x' = 0$ and $f_y' = 0$) and the values of x and y are determined from the readings of counters 5 and 6. Integrators 7 and 8 serve for generation of the partial increments $f_x'\,dx$ and $f_y'\,dy$; the sum of these two partial increments gives the total increment of the function $df = f_x'\,dx + f_y'\,dy$. At the extreme point the total increment df is equal to zero.

Let us discuss the case where the function $f(x, y)$ of a given process is unknown but the values of the partial increments may be measured. Assume (Fig. 68) that there exists at the input and output a switching arrangement which at one moment t_1 controls independent variable x (y remaining constant) and at another moment t_2 controls variable y (x remaining constant). In this case, the partial increments df_x and df_y of the unknown function $f(x, y)$ appear at the output of the process. By means of a servo integrator 1 and integrator 3, a partial derivative f_x' is obtained. The increment df_x' which is obtained by integration of $f_x'\,dx$ is compared with the actual increment appearing at the output of the process. It is assumed here that the known initial value of the partial derivative is established in the register of integrator 3. This initial value may be found by "generating" its value over a certain period of time. The output of servo integrator 5 gives the increment pulses of independent variable dx, which is fed back until the extreme point is found at which $f_x' = 0$. In a similar manner, f_y' is generated and the equation $f_y' = 0$ is solved. Thus, in order to determine

the partial derivatives, the process is performed alternately for x only and for y only over a period of time sufficient to determine the partial derivatives.

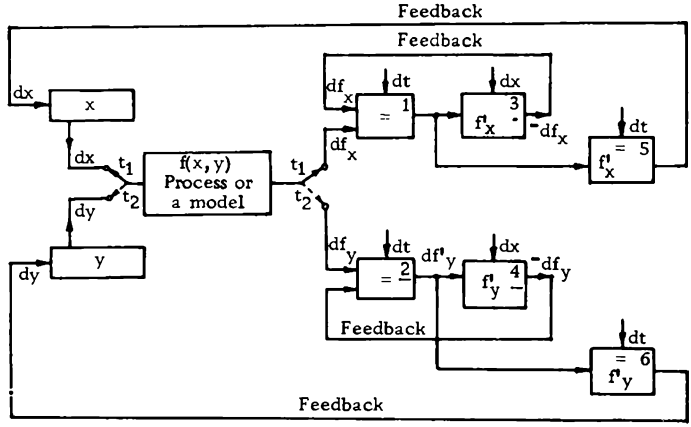


Fig. 68. Block diagram for determining extreme values of a function not given analytically.

9. LIMITING OPERATION

In the solution of some problems it is necessary to limit the change of the variable of a function to a given interval (Fig. 69a). For this purpose a servo integrator is used as the function limiter. The limiting integrator (Fig. 69b) operates as follows. When the number in the y register reaches the largest positive or negative value (i.e., $+1$ or -1), an overflow occurs and the number in the y register changes the sign. In addition, there is a change in the sign of increments ΔS at the output of servo integrator 1 and there is an interruption of application of increments to the counter of the y register of servo integrator 2 wherein these increments are summed. If the upper limit of function $y = f(x)$ is to be applied, the number $1 - A$ is first entered into the register, where $1 - A$ (Fig. 69b) is the value of y at point a and 1 is the maximum capacity of the y register.

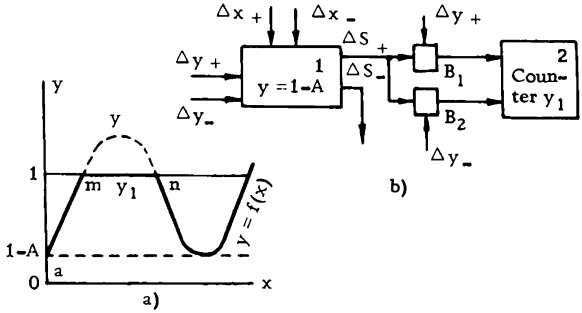


Fig. 69. Function limiter.

When the sum of positive increments Δy accumulated in the y register is equal to A, overflow of the y register of integrator 1 occurs. The number code in the y register is then equal to the largest negative number and gates B_1 and B_2, passing the positive or negative increments Δy to the y_1 counter, are closed in the region above the line mn. For any change of y above line mn (indicated by the dashed curve) the number y is always negative, first increasing in absolute value (for a positive increment of Δy) and then decreasing to zero. At point n the number in the y register again becomes equal to the largest negative number. With a further change in y below line mn an overflow of the register occurs and the number again becomes positive. In this case, gates B_1 and B_2 are open (conducting) and the y value varies along the curve below line mn. The lower level of the curve $y = f(x)$ is similarly limited if the number $-1 + A$ is entered into the y register, and negative pulses of increments ΔS are used for control of gates B_1 and B_2.

Figure 70 shows the block diagram of a limiting operation with the use of a servo integrator in the ternary system (see Chapter I, Sec. 6). The limiting gates B_1 and B_2 are connected to the output of the $\sum_1$ adder through flipflop FF and the gates which generate increments ΔS_+ or ΔS_- in accordance with the pre-established rules.

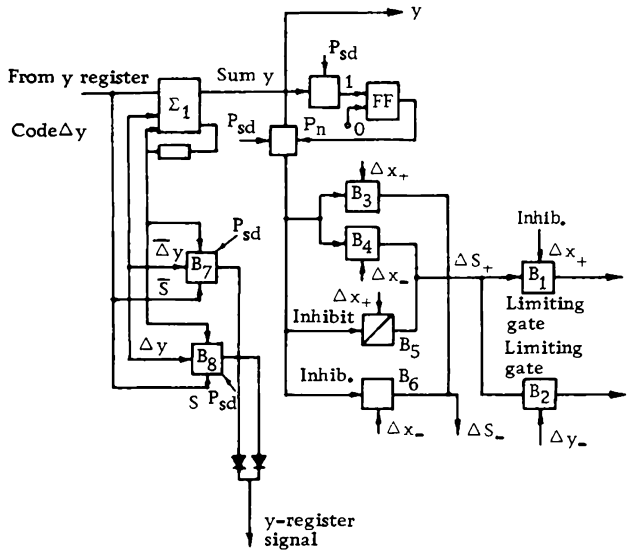


Fig. 70. Block diagram for limiting operation.

For example, assume that the number in the y register corresponding to point a (Fig. 69) is equal to $\boxed{0}$ 1000 (+8 for a register capacity of 4 bits). Then at point m the y code becomes

$$\begin{array}{r} 0\ 1000\ (1-A) \\ +\ 0\ 1000\ (+A) \\ \hline \boxed{1}0000 \end{array}$$

With a further increase in y, the number in the register above line mn remains negative, taking the values $\boxed{1}$ 0001, $\boxed{1}$ 0010, etc. At point n, the number code of y is again $\boxed{1}$ 0000. With a further decrease in y ($\Delta y = \boxed{1}1111$) an overflow occurs from the y register and its code again becomes positive.

Provision is usually made to stop the computer upon overflow of the y register of any integrator with the exception of servo integrators used for limiting and certain other operations. Gates B_7 and B_8 are used in ordinary integrators for detecting positive or negative overflow (see Chapter I, Sec. 6). In our example, the overflow pulse of the y register must be blocked in order to avoid interruption of computer operation.

Simpler limiting circuits are obtainable when binary code (+1 or −1) is used in conjunction with an ordinary servo integrator (see Chapter I, Sec. 10). For example (Fig. 71a), the servo integrator yields positive increments ΔS as long as the quantity $1 - A$ in the y register remains positive. In addition, the positive increments accumulated in the y_1 counter are equal to $2 \cdot \Delta y_1$, since increments of argument Δx arrive simultaneously at the increment adder $\sum_\Delta$. When $1 - A$ is negative (above line mn, Fig. 71c), increments ΔS become negative and increments Δx remain positive. Hence the increment at the $\sum_\Delta$ output is equal to zero until the number in the y register changes sign to a positive value.

Another version of the servo integrator limiter (Fig. 71b) uses the Δy increments at the independent variable input of the integrator.

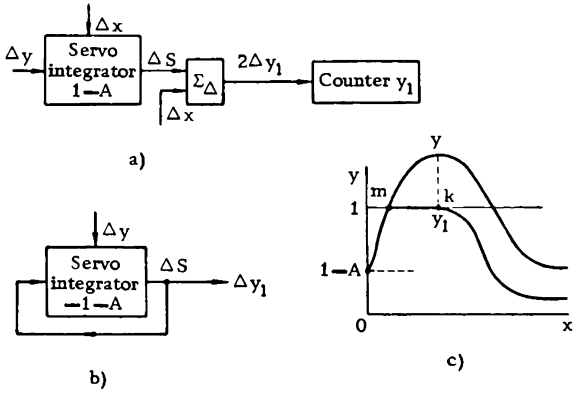


Fig. 71. Limiters with servo integrator.

The ΔS increments from the integrator output are fed back to the input. With positive values of Δy, the ΔS increments are also positive and increase the number in the y register to the maximum positive value; thereafter an overflow of the register occurs and the code changes to the maximum negative number. The result at the integrator output is the zero rate of increments (+1 −1 +1 −1, etc.) from point m to point κ (Fig. 71c). When the y increments (below point k) become negative, the sign of the ΔS overflow changes and the absolute value of the negative number in the integrator register decreases. As a result, a sequence of negative increments of ΔS appears at the output corresponding to the portion of the curve below point k. As is seen from Fig. 71c, limiting of y occurs only between points m and k and the portion of the curve beyond point κ is similar to the given curve but does not coincide with it.

10. LOGICAL "SIGN" OPERATION

The "sign" operation controls the sign of the increments at the output of a second integrator according to the sign of the number code in the y register of a first integrator. For example, if the sign of the number code in the y register of the first integrator is positive (code 0), the sign of the increments from the output of the second integrator remains unchanged, (i.e., as in the case of summation). However, if the sign of the number code in the y register of the first integrator is negative (code 1), the sign of the increments from the output of the second integrator is reversed. This operation is used in controlling the process of problem solution according to the obtained result of the solution (i.e., similar to those logical operations in conventional digital computers known as the conditional or unconditional transfer operations). In addition, the sign operation is used in the generation of a number of special functions.

Let us discuss two examples of the use of the sign operation. The first example (Fig. 72) is the conversion of a sign-alternating function $f(x)$ to a function $f'(x)$ which has the absolute value of function $f(x)$. Here there occurs, as it were, full-wave rectification of a sinusoidal function.

Assume that one integrator stores in the y register the sign-alternating integrand $f(x)$. The sign digit (sign code) of this function establishes code 1 or 0 in the last (highest) S register of a second integrator. If the function $f(x)$ of the first integrator is positive, the overflow output dS of the second integrator remains unchanged (code 0 is set in the highest digit of the S register of the second integrator). If the integrand $f(x)$ of the first integrator is negative, the overflow output dS of the second integrator reverses sign (i.e., if the output has code 0, the code becomes 1 and vice versa).

Integrator 1 in Fig. 72 serves for integration of function $y' = f'(x)$; at its output appear the increments $dy = df(x)$ of the

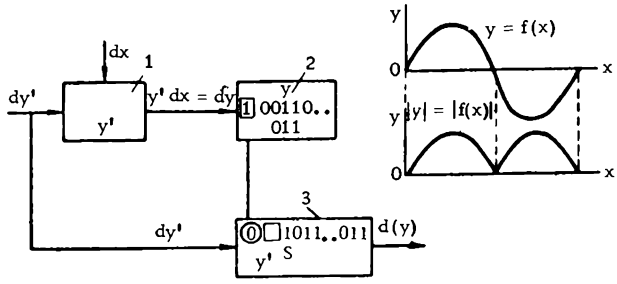


Fig. 72. Conversion of a sign-alternating function $f(x)$ to a function $|f(x)|$.

sign-alternating function. Increments dy are accumulated in the y register of integrator 2. The sign code of the y register controls the code of the last position of the S register of integrator 3 in such a manner that code 1 in the sign location of y corresponds to code 0 in the last position of the S register of integrator 3 and vice versa. Thus, when the sign of $f(x)$ is positive, the code of overflow pulses dS of integrator 3 does not change. Since increments dy are also applied to integrator 3, its output gives the increments of function $f(x) = dy$. If the sign code of $f(x)$ in integrator 2 is zero ($f(x)$ is a negative function), the overflow code at the output of integrator 3 undergoes sign reversal and, consequently, increments of the function $-[-f(x)]$ (i.e. $+f(x)$) appear at the output of integrator 3.

Thus, at the output of integrator 3 in both cases there are generated increments of function $f(x)$ always having positive value, i.e.:

a) if $f(x) \geqslant 0$, the output of integrator 3 is $+f(x)$;

b) if $f(x) \leqslant 0$, the output of integrator 3 is $+f(x)$;

The second example shows the use of such integrators for the solution of differential equations:

$$\frac{d^2y}{dt^2} + k_1 y = 0, \quad \text{if} \quad y \geqslant 0, \tag{117}$$

or

$$\frac{d^2y}{dt^2} + k_2 y = 0 \quad \text{if} \quad y \leqslant 0. \tag{117a}$$

Depending on the sign of the unknown function of y, the problem solution requires transition from one equation with coefficient k_1 to the other with coefficient k_2.

The given problem is solved by means of the diagram shown in Fig. 73. Depending on the sign of the integrand $\frac{d^2y}{dt^2}$ of integrator 1, there is a change in the code of the last position of the S register

of integrator 2 and in the code of the overflow pulses at its output. The equations may be written in the form

$$\frac{d^2y}{dt^2} = -k_1 y \quad \text{or} \quad \frac{dy}{dt} = -k_1 \quad \text{for} \quad y \geqslant 0, \tag{118}$$

$$\frac{d^2y}{dt^2} = -k_2 y \quad \text{or} \quad \frac{dy}{dt} = -k_2 \quad \text{for} \quad y < 0. \tag{119}$$

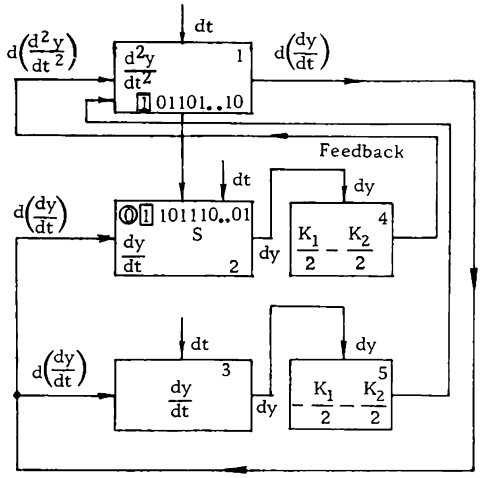


Fig. 73. Diagram for solving a differential equation with automatic change of coefficients.

In adopting the arrangement in Fig. 73, $\frac{d^2y}{dt^2}$ is assumed known. Integrator 1 gives $d\left(\frac{dy}{dt}\right)$ and integrator 2 (or 3) gives increment dy. The scaling integrators 4 and 5 perform multiplication of increment dy by constant factors: integrator 4 by coefficient $\frac{k_1}{2} - \frac{k_2}{2}$ and integrator 5 by coefficient $-\frac{k_1}{2} - \frac{k_2}{2}$.

If $y \geqslant 0$ (i.e., $\frac{d^2y}{dt^2}$ is negative), the code of the last digit of the S register in integrator 2 is 1 and the overflow pulse of integrator 2 reverses the usual code, i.e., the output of integrator 4 gives $-dy\left(\frac{k_1}{2} - \frac{k_2}{2}\right)$. As the result of summation of the increments of integrators 4 and 5, we obtain an increment at the input of integrator 1

$$d\left(\frac{d^2y}{dt^2}\right) = -dy\left(\frac{k_1}{2} - \frac{k_2}{2}\right) + dy\left(-\frac{k_1}{2} - \frac{k_2}{2}\right) = -k_1 dy$$

or $d\left(\frac{d^2y}{dt^2}\right) = -k_1 dy$. In this case the problem solution is achieved from equation (117).

If $y < 0$ (i.e., $\frac{d^2y}{dt^2}$ is positive), the code in the last S register of integrator 2 is 0 and the overflow pulse of this integrator has the usual code. In this case the sum of the increments of integrators 4 and 5 is

$$d\left(\frac{d^2y}{dt^2}\right) = \left(\frac{k_1}{2} - \frac{k_2}{2}\right)dy + \left(-\frac{k_1}{2} - \frac{k_2}{2}\right)dy = -k_2 dy,$$

i.e., solution of the problem is achieved from equation (117a).

The sign operation in a DDA is achieved on the basis of the diagram in Fig. 74. Through gate B_1, the sign digit of the number code y from the y register of the first integrator arrives at the input of flipflop FF and is stored therein until readout to the second integrator. The necessity for execution of the sign operation is indicated by code 1 in the second control digit of the S register of the first integrator (or in the second control digit of the y register of the first integrator if it is not occupied for other control purposes). The entered code for the sign operation arrives through gate B_2 upon passage of digit pulse P_2 and is stored in flipflop FF_2.

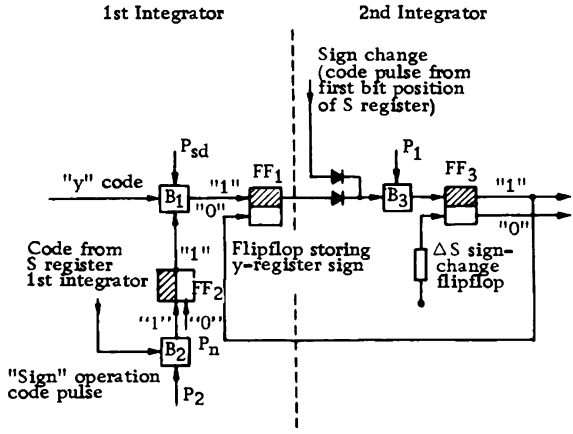


Fig. 74. Logical sign operation.

Flipflop FF_1, storing the sign of y of the first integrator, opens gate B_3 through which digit pulse P_1 sets flipflop FF_3 of the second integrator to the code 1 state. Flipflop FF_3, in turn, controls the circuit for sign change of the ΔS increments of the second integrator (see Sec. 9). When flipflop FF_3 has been set to the 1 state, the signal from its output resets flipflop FF_1 to the "0" state. Flipflop FF_3 will be set to zero after the passage of the last pulse P_n in the second integrator. The above described scheme may be repeatedly used for the sign operation in several pairs of integrators during one integration cycle.

11. SOLUTION OF INTEGRAL EQUATIONS

The digital integrating computer is also useful in the solution of integral equations. Some problems in science and engineering applications can be reduced to the solution of the Fredholm-Volterra integral equation. The Fredholm equation has the form

$$F(x) - \varphi(x) = \int_a^b K(x, S)\varphi(S)\,dS, \tag{120}$$

where K and F are known functions and φ is an unknown function. The limits of integration a and b are the limits of change of independent variable x as well as of independent variable S.

Solution of the integral equation is based on the method of successive approximation or the iteration method. The iteration method is discussed in detail in numerical analysis. For solution of the Fredholm equation we may use the method of simple iteration and the Gauss-Siedel method [15].

For simple iteration

$$F(x) - \varphi^{(k)}(x) = \int_a^b K(x, S)\varphi^{(k-1)}(S)\,dS. \tag{121}$$

For the Gauss-Siedel iteration

$$F(x) - \varphi^{(k)}(x) = \int_a^{x-\Delta} K(x, S)\varphi^{(k)}(S)\,dS - \int_{x-\Delta}^b K(x, S)\varphi^{(k-1)}(S)\,dS, \tag{122}$$

where the parenthesized superscript denotes the number of the iterative step; Δ is a sufficiently small quantity.

The Gauss-Siedel iteration differs from simple iteration in that, in computing the sought function, we use new values as soon as they are calculated in an iterative step, whereas in simple iteration we use the values which have been obtained in the previous iterative step. Thus, renewal of data in Gauss-Siedel iteration occurs continuously and that in simple iteration only upon the completion of an iterative step.

Application of the Gauss-Siedel method reduces to approximation of the integral equation as a system of algebraic equations which is solved in sequence by the iteration method.

In numerical solution of the Fredholm equation by means of a DDA, equation (120) may be written in the differential form

$$dF(x) - d\varphi(x) = \int_a^b K(x, S)\cdot\varphi(S)\cdot dS. \tag{123}$$

With the given initial values for the unknown function, we may assume at the beginning of each iteration that function $\varphi(S)$ is known.

Then by means of integrator 1 (Fig. 75) we obtain the increment $\varphi(S)\,dS$ (which is used as the argument for integrator 2). At the output of integrator 2 we obtain the increment $K(x, S)\,\varphi(S)\,dS$ if the input to the integrand of this integrator is the increment of known function $K(x, S)$. We note that differential $dK(x, S)$ is a function of two independent variables as below:

$$dK(x, S) = \frac{\partial K(x, S)}{\partial x}\,dx + \frac{\partial K(x, S)}{\partial S}\,dS. \tag{124}$$

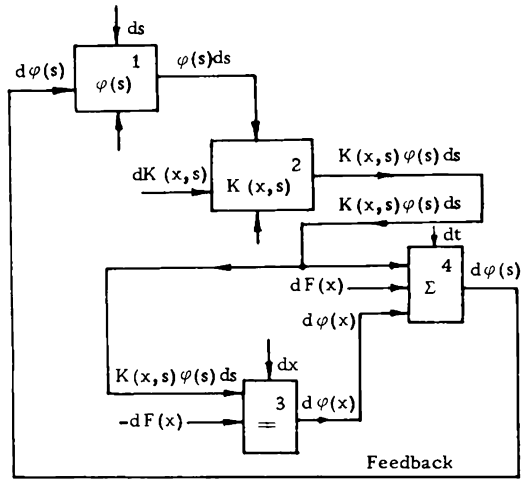


Fig. 75. Diagram for solving integral equation (120).

From equation (123), the increment of unknown function $\varphi(x)$ must be

$$-d\varphi(x) = K(x, S)\,\varphi(S)\,dS - dF(x). \tag{125}$$

If the right side of equation (125) is not equal to zero, then there must occur an increment of function $\varphi(x)$. Generation of the increment $d\varphi(x)$ is performed by servo integrator 3 where increments $K(x, s)\,\varphi(S)\,dS$ and $-dF(x)$ are applied to its input. In the y register of the servo integrator the difference of increments $K(x, S)\,\varphi(S)\,dS$ and $-dF(x)$ is accumulated. As long as this difference is not equal to zero, positive or negative increments $d\varphi(x)$ appear at the output of the servo integrator, depending on the sign of the difference. Increment $d\varphi(x)$ occurs at the output of servo integrator 3 only in the presence of a unit increment dx. Herein the sign of the unit increment $d\varphi(x)$ at the output of the servo integrator is determined by the sign of the product of the number code in the y register (i.e., the difference $K(x, y)\,d\varphi(S)\,dS - dF(x)$) and increment dx. The output of servo integrator 3, increment $d\varphi(x)$, is applied to the input

of the summing (servo) integrator 4, where the following summation is performed

$$K(x,\ S)\,\varphi(S)\,dS - dF(x) + d\varphi(x) = 0. \tag{126}$$

If the algebraic sum of these increments stored in the y register of integrator 4 is not equal to zero, an increment of one or the other sign is generated at its output and is fed back negatively to the input of integrator 1 as the increment of the unknown function $\varphi(S)$. The initial values of the integrands $\varphi(S)_{\text{init}} = \varphi(a)_{S=a}$ and $K(x, S)_{\text{init}} = K(x, S)_{\substack{x=a \\ S=a}}$ must be given to integrators 1 and 2. For $x = a$ and $S = a$, the initial values in the registers of servo integrators 3 and 4 are taken equal to zero (or, more precisely, the code $\boxed{0}$ 000 00, with which the increment at the output of the servo integrator is equal to zero).

If the known function $K(x, y)$ of two variables is given in the form of tables or curves, function generation of such a form may be obtained by the methods to be discussed in Chapter VIII, Sec. 4.

Consider the solution of an integral equation which has the form:

$$\gamma_x = 0.25 + \int_0^x \frac{\Phi(x)\cdot y(x)\,[1+0.14\,y(x)]\,[1+0.02752\,y(x)]}{[1-y(x)]\,[1-\Phi(x)\cdot x]}\,dx, \tag{127}$$

where $\Phi(x)$ is the unknown function;

γ_x, $y(x)$ are given functions of x.

It is required to find the value of $\Phi(x)$ for which $y = 1$ and $x = 1$.

Equation (127) may be written in a differential form

$$d\gamma(x)\cdot[1-y(x)]\,[1-\Phi(x)\cdot x] = \Phi(x)\cdot y(x)\,[1+0.14\,y(x)]\times \\ \times\,[1+0.02752\,y(x)]\,dx \tag{128}$$

or

$$d\gamma_x\cdot[1-y(x)-\Phi(x)\cdot x+\Phi(x)\cdot x\cdot y(x)] = \\ = \Phi(x)\cdot y(x)\cdot[1+0.14\,y(x)]\cdot[1+0.02752\,y(x)]\,dx. \tag{129}$$

The setup for solution of this equation is shown in Fig. 76, requiring 18 integrators. The initially known approximate value of $\Phi(x) = 0.3$ is used as the initial value in integrators 1, 6, 10, 13 and 16. The initial value of y is 0.25 for $x = 0$. The computed values of x for $y = 1$ are printed out automatically in the form of a table at the output of the integrating computer. According to the change in x, the corresponding values of $\Phi(x)$ are computed and printed out in tabular form. The results of solution of the integral equation are given in a graphic form in Fig. 77. Each computed approximate solution for $\Phi(x)$ is again automatically inserted into the computer in order to obtain the next more precise solution. The final value of $\Phi(x)$ at point ($x = 1$ and $y = 1$) is approximately 0.2563. A discontinuity in the function $\Phi(x)$ occurs at $x = 1$.

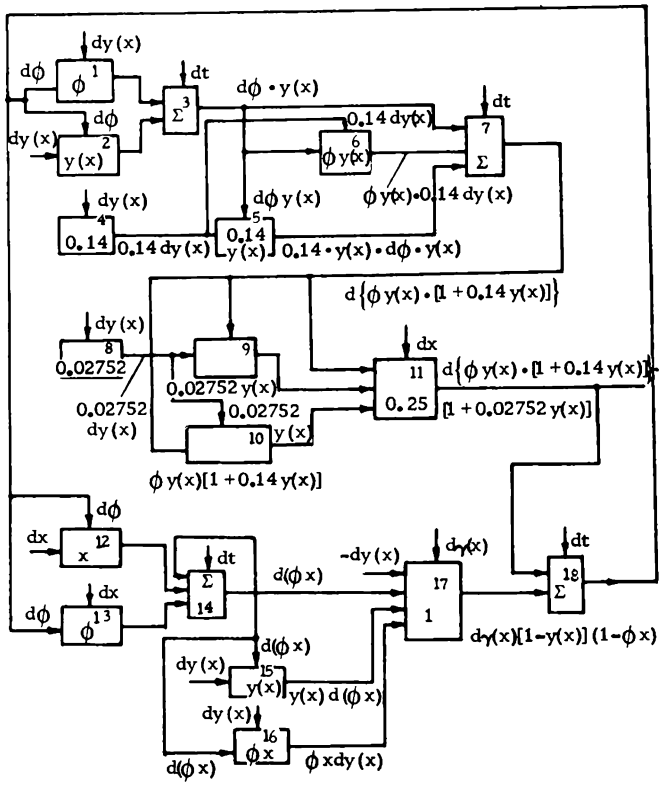


Fig. 76. Block diagram for solving integral equation (127).

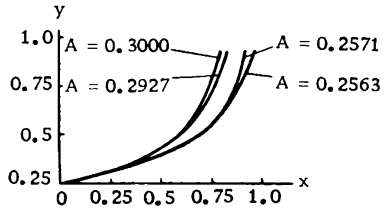


Fig. 77. Solution of integral equation (127).

12. CORRELATION ANALYSIS

Many problems in research and development are solved by statistical methods. A typical problem is the analysis of automatic control systems subject to random disturbances or noise. One experimental method of determining the behavior of such a system

is to apply random disturbances or signals of a given waveform to its input and examine the resulting output signals by means of correlation analysis.

In these cases, the computer is required to calculate one of the statistical characteristics of the random processes, the so-called correlation function,

$$R(\tau) \approx \frac{1}{T} \int_0^T f(t) \cdot f(t+\tau)\, dt, \tag{130}$$

where $f(t)$ is the given function of a random process and is recorded on an oscillogram or paper tape;
T is the duration of process observation;
τ is a parameter of time delay.

Machine computation of the correlation function $R(\tau)$ requires that τ can be varied. The computation of integral (130) is repeated as many times as the chosen values of τ so as to give enough points of correlation function $R(\tau)$.

To compute the correlation function in a computer, the function $f(t)$ of a random process is given in the form of an oscillogram or a graph. The oscillogram recording is reproduced as a plot of $f(t)$ which is converted by a function generator (see Chapter VIII, Sec. 2) into increments of $\Delta f(t)$ and stored on punched tape. Two punched tapes (Fig. 78) are prepared and inserted into the computer; the tape speed is chosen according to the required time scale. The delay time τ in formula (130) is achieved by displacing one punched tape relative to the other by a segment proportional to τ. By means of integrators 1 and 2 multiplication of $f(t) \cdot f(t+\tau)$ is performed in the computer. Integrator 3 and register 4 of integrator 4 give the quantity $\int_0^T f(t)\, f(t+\tau)$. Before computation begins, the initial values of functions $f(t)$ and $f(t+\tau)$ must be inserted into integrators 1-4; these values may be stored in incremental form on the section of tape preceding the record of the variables of $f(t)$ and $f(t+\tau)$.

The simplest method of recording random processes is the generation of increments of functions $f(t)$ and $f(t+\tau)$ by means of digital transducers installed on the investigated objects. Thus, there is no need of units for conversion and insertion of these functions into the computer.

Many problems in correlation analysis may be expressed analytically in the form of systems of differential or integral equations which are machine solved by integration methods. Among these problems are those of automatic control. For example, if the inputs or disturbances to the system are stationary random processes described by appropriate correlation functions $R(\tau)$, the minimum mean square error is defined by a Fredholm equation of the first kind with a convolution type kernel

$$\int_0^{\infty} R_{\varphi}(\tau - \lambda)\, k(\lambda)\, d\lambda, \tag{131}$$

where $k(\lambda)$ is the impulse transfer function of the system.

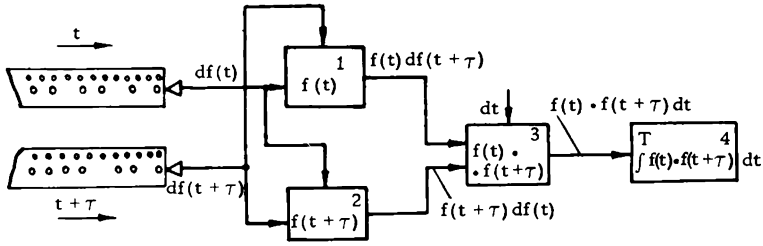


Fig. 78. Block diagram for computing a correlation function.

If the inputs or disturbances are nonstationary random processes described by correlation functions $R(t, \tau)$, the minimum mean square error is defined by an equation of the form

$$\int_0^{T} \sigma(t - \theta)\, \varrho(\tau - \theta)\, k(\theta, \tau)\, d\theta, \tag{132}$$

where $\varrho(\tau)$ is the correlation function of a stationary process; $k(\theta, \tau)$ is a given function of two parameters τ, θ. Thus, the solution of this problem is reduced to Fredholm equations of first kind with correlation functions as the kernels.

13. PROGRAM PREPARATION

The program for DDA operation specifies the interconnections of the integrators with due consideration of scale relations. These interconnections are given not only for the integrators required for problem solution, but also for scaling integrators, integrators for interpolation of the given functional dependences, and integrators for smoothing and generation of the input parameters. Functions which may be obtained by the solution of differential equations are also included in the setup diagram. In preparing the program the initial values of the integrands of all integrators should be determined and inserted into the proper integrators. In choosing scale factors it is necessary to determine the possible maximum values of all variables and to code the proper position of the initial pulse H for the y register of each integrator, which defines the "value" of the unit increment.

In order to obtain optimum accuracy of solution, in preparing the setup diagram it is necessary to avoid the differentiation operation since this operation is considerably less accurate than integration. When derivatives are required, a single differentiation

operation, apart from the rest of the problem, is performed and the resulting error corrected. Accuracy is increased by using as much of the capacity of each register as possible (at the maximum value of the given quantity) and by decreasing the rate of the dependent variable (rather than the rate of the independent variable).

From the setup diagram a table of A_x and A_y is prepared, indicating the locations where increments are applied to the x and y inputs of each integrator and where each integrator output or each input channel is connected.

The basic parameter of time t is inserted into the computer in the form of clock pulses CP, and the functions recorded on tape are entered in the form of increments.

The output of only one integrator may be applied to the x input of an integrator; the outputs of several integrators may be applied the the y input.

For preparation of the coded program of A_x and A_y, all locations in the overflow lines are given pulse numbers from P_1 to P_{17} (assuming 18 digit positions in the register). One or two digit pulses are usually allotted to control and thus cannot serve as locations for storage of the program codes. In determining the overflow position of any integrator in a current overflow line, it must be kept in mind that its position (pulse number) varies in a one-bit-short delay line (Chapter V, Sec. 8) or in a shift register. Therefore, it is necessary to determine the position (digit number) where the required overflow is located with consideration of the displacement of the overflow by one digit for each integrator.

Written in this form, the A_x and A_y program shows in which integrators the code must be 1. If in preparing the program two overflow pulses occur at the same time, it is necessary to alter the integrator connection sequence in such a manner that the entry of these overflows is distributed in time. For long-term storage of overflows and for storage of increments from input channels, several storage lines are used. The codes from the current overflow line are selected according to the A_{y1} program. The first two codes of the A_x program (P_1 and P_2) serve for selection at the x input from any line $\Delta S_1 - \Delta S_5$ while the first two codes of the A_y program serve for entry of the current increments in a long-term overflow storage line (ΔS_2 and ΔS_3; see Chapter V, Sec. 8, 9).

The A_{y_2} program is designed for selection of increments at the y input of the integrator from the long-term overflow storage line or from the increment storage line of the input channels. The first two codes of the A_{y_2} program (corresponding to pulses P_1 and P_2) serve for selection of one of these lines.

In the presence of two A_{y_1} and A_{y_2} programs the A_y program must be broken down into two parts and coded separately for the current overflows and for long-term overflow storage. Supplementary A_y programs may be required for write-in and read-out of increments from the input channel as well as for increasing the value of the increment storage.

Chapter III

Digital Computer Circuits

1. SWITCHING OPERATION OF THE JUNCTION TRANSISTOR

The collector current I_c of a junction transistor is

$$-I_c = I_{c_0} + \alpha I_e, \tag{1}$$

where α is the current gain ($\alpha = 0.9\text{-}0.98$) in the grounded-base configuration ($\alpha \approx \frac{I_c}{I_e}$).

The emitter current I_e is the sum of the collector and base currents:

$$-I_e = I_c + I_b. \tag{2}$$

From expressions (1) and (2) we have,

$$I_c = \beta (I_b - I_{c_0}) - I_{c_0}, \tag{3}$$

and

$$\beta = \frac{\alpha}{1-\alpha}, \tag{4}$$

where β is the current gain in the grounded-emitter configuration $\left(\beta \approx \frac{I_c}{I_b}\right)$.

The transistor is cut off (i.e., $I_c = I_{c_0}$) when $I_b = +I_{c_0}$. This is achieved by applying a positive voltage $V_{be} > 0$ between the base and the emitter. If $V_{be} = 0$ then $I_b = 0$ and the collector current is

$$I_c = -I_{c_0}(\beta + 1). \tag{5}$$

With a large β, the collector current may have considerable value during the cut-off condition. The initial collector current I_{c_0} of a junction transistor increases exponentially with increase in temperature:

$$I_{c_0} = I_{c_{0\,20^\circ}} e^{0.07\,(t-20^\circ)} \approx I_{c_{0\,20^\circ}} \cdot 2^{\frac{t^\circ - 20^\circ}{10}}, \tag{6}$$

where $I_{c_0 20°}$ is the initial collector current at temperature 20°C. Thus, the initial collector current I_{c_0} is approximately doubled with each temperature increase of 10°C.

The initial collector current I_{c_0} of a junction transistor is marked by considerable "creep" in time. The larger the collector voltage of the cut-off transistor, the larger the "creep" of I_{c_0} in time.

The collector characteristic of a transistor (Fig. 79a), showing the relation between the collector current I_{c_0} and collector voltage V_c (with the base current I_b as the parameter), resembles the characteristic of a pentode (i.e., the collector current is almost independent of collector voltage). The parameter of the family of curves, the base current I_b, is a principal feature of the transistor.

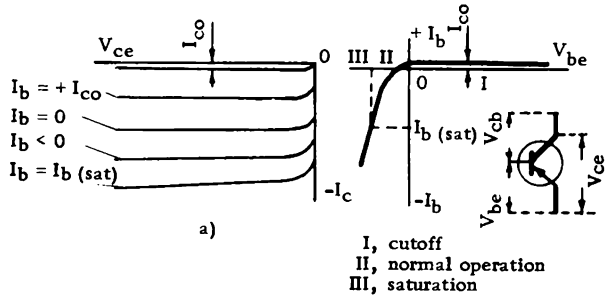


Fig. 79. Junction transistor characteristic: a—collector characteristic; b—base current as a function of base voltage.

The junction transistor can operate in one of the three regions shown in the characteristic (Fig. 79b):

a) Cutoff region, where the base-to-emitter voltage is positive $V_{be} \geqslant 0$ while the collector-to-base voltage V_{cb} is negative.

With a small but positive voltage at the base, the collector current decreases negligibly and, at voltages greater than +0.05 v, remains almost constant at 0.01—0.03 ma. With the current $I_b = 0$ there is a small initial collector current I_{c_0} and the transistor is in the cut-off state. The d-c resistance of a cut-off transistor ranges from several kilohms to one or two megohms.

b) Linear region, where the base-to-emitter is negative $V_{be} < 0$ and the transistor conducts. The collector-to-base voltage is negative.

c) Saturation region, where the base-to-emitter voltage is negative ($V_{be} < 0$ for p-n-p transistors) and the base current is relatively large. A transistor operates in the saturation region when the base current I_b exceeds a certain value and the collector current is limited by a load resistance R_L. With a given value of load resistance R_L in the collector circuit, in order to establish the collector saturation current in the vicinity of $-\frac{E_c}{R_L}$ (where E_c is the collector supply voltage) the base current is

$$I_{b(sat)} = \frac{\frac{E_c}{R_L} + I_{c_0}}{\beta} + I_{c_0}, \tag{7}$$

which is obtained by insertion of $I_c \approx -\frac{E_c}{R_L}$ in equation (3). Thus, over a wide range of temperatures (including low temperatures when current I_{c_0} is small), a junction transistor remains saturated when the absolute value of base current is larger than or equal to $E_c/R_L\,\beta$, i.e.,

$$|I_{b(sat)}| \geqslant \frac{E_c}{R_L \beta}. \tag{8}$$

In the saturation region the transistor may remain in the conducting state for a certain period of time after removal of the signal in the base circuit. This is due to the fact that upon saturation of the transistor there occurs an excess concentration of minority carriers in its base. Since the collector potential is low during saturation, the recombination of minority carriers after removal of the signal in the base circuit prolongs for a certain time the low collector potential. The turn-off time of the transistor is approximately 2-3 times longer than its turn-on time. During saturation, the emitter-collector voltage V_{ce} (as well as V_{be} and V_{cb}) is extremely small—of the order of 0.05-0.1 v. Moreover, the d-c resistance in the collector circuit is of the order of 1-25 ohms and is a fraction of an ohm in power transistors. Since the voltage drop between any two terminals of a transistor is small, in some cases the saturated transistor may be regarded as a nodal point with the same potential.

The switching operation of transistors is relatively independent of temperature and the parameters of the transistors need not be exactly matched. If a transistor is switched into saturation by a signal calculated for minimum gain, other transistors will ensure normal operation of the circuit.

It must be kept in mind that the β current gain for a common-emitter circuit in the above formulas depends on the collector current I_c (Fig. 80). With an increase in I_c, the β gain decreases and at large collector currents it may be of the order of several units (less than 10). In designing switching circuits, it is necessary to determine the value of collector current from the collector characteristic and to determine the β gain from the curve in Fig. 80 (for the given type of transistor).

Let us now discuss one method of calculating the stable states of the flipflop circuit (Fig. 81a) with bias $+E_B$ at the base [20]. Assume that the left transistor T_1 is cut off and the right transistor T_2 is in saturation. Then, the potentials at points a and b are approximately zero since V_{ce} and V_{be} for a saturated transistor are tenths or even hundredths of a volt. The base-emitter voltage of the off-transistor T_1 (Fig. 81b) is

$$V_{be} = \frac{R_1}{R_1 + R_2}(E_B - I_{c_0} \cdot R_2), \tag{9}$$

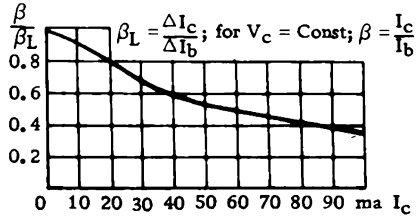


Fig. 80. Ratio β/β_L as a function of collector current I_c.

since the current $+I_{c_0}$ flows through the base; the initial collector current is $-I_{c_0}$, while the emitter current $I_e = -(I_c + I_b)$ is close to zero. The cutoff of transistor T_1 requires that the base-emitter voltage be greater than or equal to zero, i.e., that

$$V_{be} \geqslant 0.$$

From expression (9), we have

$$\frac{R_1}{R_1 + R_2}(E_B - I_{c_0} \cdot R_2) \geqslant 0. \tag{10}$$

This means that the quantity in the parentheses must be greater than or equal to zero, i.e.,

$$\frac{E_B}{R_2} \geqslant I_{c_0}. \tag{11}$$

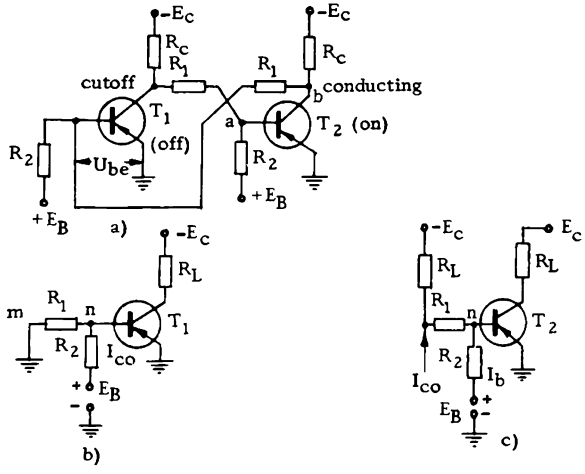


Fig. 81. Transistor switching circuits.

Condition (11) for cutoff of transistor T_1 must also be satisfied at the highest temperature of the operating range for which I_{c_0} has the maximum value $I_{c_0\,max}$. In this case, we may proceed from inequality (11) to the following equality:

$$R_2 = \frac{E_B}{I_{co\,max}}. \tag{12}$$

For the saturated transistor T_2, the base current I_b (absolute value) is

$$|I_b| = \frac{E_c - I_{c_0} R_L}{R_1 + R_L} - \frac{E_B}{R_2}. \tag{13}$$

The above base current (at point n in Fig. 81c) consists of two components. The first component is determined by the voltage $-E_c$ and current through resistors R_L, R_1 and the emitter. Herein it is necessary to consider the voltage drop due to current I_{c_0} of the off-transistor. From the first component we must subtract the current passing from source $+E_B$ through resistor R_2 and the emitter in the reverse direction, which is equal to $\frac{E_B}{R_2}$. Hence current I_b is determined from formula (13).

The collector current of the saturated transistor is,

$$I_{c\,(sat)} = \frac{E_c}{R_L}. \tag{14}$$

This current must be many times greater than the maximum value of initial collector current $I_{c_0 max}$, or

$$I_{c_0\,(sat)} = \frac{E_c}{R_L} \gg I_{c_0\,max}. \tag{15}$$

From equation (13), with expression (15) in mind, we obtain

$$|I_b| = \frac{E_c}{R_1 + R_L} - \frac{E_B}{R_2}. \tag{16}$$

Current I_b for the saturated transistor must be greater than or equal to $\frac{E_c}{\beta \cdot R_L}$ (8), i.e.,

$$\frac{E_c}{R_1 + R_L} - \frac{E_B}{R_2} \geqslant \frac{E_c}{\beta \cdot R_L}, \tag{17}$$

whence we find that

$$R_1 < \left(\frac{\beta}{1 + \beta \cdot \frac{E_B}{E_c} \cdot \frac{R_c}{R_2}} - 1 \right) R_L. \tag{18}$$

Formulas (12) and (18) permit us to calculate the divider resistances R_1 and R_2 of the flipflop circuit if we are given voltages E_c and E_B. Resistance R_L determines the collector current of the conducting transistor.

It is noted that the ratio R_1/R_2 is of importance in a vacuum-tube flipflop and in a semiconductor flipflop. Values of resistors R_1 and R_2 are also of importance. Amplitude of the output voltage pulse at the collector of the transistor is

$$V_{out} = \frac{R_1}{R_1+R_L}(E_c - I_{c_0}R_L). \tag{19}$$

In calculating resistor R_1 of the divider, the β gain must be taken at the minimum value, and it is necessary to take the value of β measured at collector current $I_{c_{(sat)}} = E_c/R_L$.

2. INVERTER AMPLIFIERS

The inverter amplifier is a principal circuit in a transistorized computer. It provides two fixed levels of output voltage corresponding to the codes 0 and 1. In the absence of input voltage (code 0) transistor T (Fig. 82a) is cut off by bias voltage +12 v, which creates a current in divider R_1 and R_2. At the transistor base (at point m) there is then obtained a potential of approximately +1.8 v relative to the grounded emitter, with the result that the transistor is adequately cut off. If a negative voltage of −6 v is applied to the input, the potential of point m at the transistor bases is then −3 v and the transistor conducts heavily. In addition, the output potential changes from −12 to 0 v. Upon removing the input voltage (from −6 to 0 v) the transistor is rapidly cut off, since a cutoff voltage of +1.8 v is again established at the base. In the presence of a large signal at the input, the transistor becomes saturated with the result that an excess concentration of minority carriers occurs in the base. The disappearance of excess concentration by recombination occurs relatively slowly after the transistor is turned off. For this reason, the turn-off time of the transistor is usually 2-3 times longer than the turn-on time, and the trailing edge of the pulse at the output is "delayed." If there is an adequately large cutoff voltage at the base, the turn-off time is considerably decreased. In order to accelerate the transistor switching, capacitor C (Fig. 82b) is connected in parallel with resistor R_2. This capacitor decreases the resistance R_2 temporarily and thereby shortens the rise time of the negative voltage at the transistor base.

The low level of output voltage with the transistor at cutoff is clamped by diode D at a voltage of −6 v. Diode D decreases the pulse fall time since even before the transistor is fully turned off the voltage at the output rapidly reaches a value of −6 v.

A disadvantage of the inverter amplifier with the transistor in conduction is the deep saturation of the transistor and storage of minority carriers, which protracts the pulse width upon turn-off.

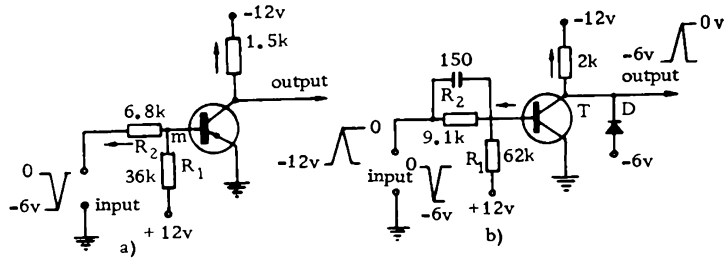


Fig. 82. Inverter circuits:
a—resistance-coupled inverter; b—inverter with accelerating capacitor.

Moreover, there is a relatively high noise level, particularly with an increase in temperature, and an increase in initial collector current I_{c_0}. With high-frequency transistors these inverters may be used as high-speed switches. Cross coupling of two inverters permits the creation of a static flipflop (see Sec. 4).

An inverter amplifier with several inputs (Fig. 83) may be used to obtain the output signal as the logical sum of several negations (NOT operation). In the absence of signals a, b, c at the transistor input, it is cut off by the bias voltage of +12 v and the output is at a potential of −12 v. If voltage a, b, or c is applied to one of the inputs, the transistor conducts and the potential at the output becomes approximately equal to zero. This circuit is simple and reliable. But a disadvantage of this circuit is the deep saturation of the transistor and the storage of minority carriers

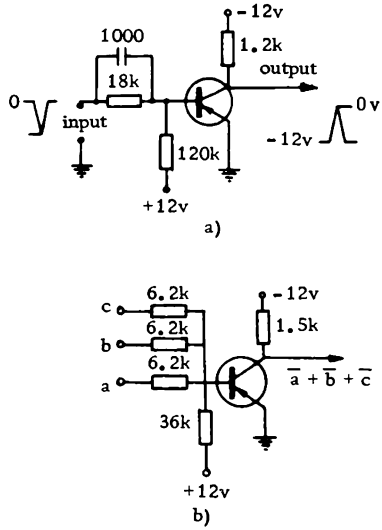


Fig. 83. Inverter amplifier:
a—single-input inverter amplifier; b—circuit with several NOT operations.

with the presence of more than one signal at the inputs, which lowers the switching rate of the transistor.

Due to the higher input resistance and power gain of the grounded-emitter configuration, the inverter amplifier is a principal element in transistor logic circuits. The use of high-frequency junction transistors (e.g., diffused junction type) permits a considerable decrease in the minority carrier recombination time, and to a certain degree eliminates the principal disadvantage of the more pronounced frequency dependence of gain in a grounded emitter circuit. It must be kept in mind that, with large input signals, the current gain β is nonlinearly dependent on emitter current and its value decreases rapidly with an increase in emitter current.

3. EMITTER FOLLOWERS

In the emitter follower (Fig. 84) the base current is the current of the input circuit and the emitter current is generated in the output circuit. It is similar to the cathode follower in vacuum tube circuits.

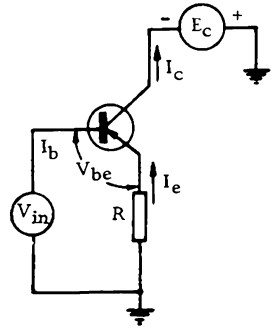


Fig. 84. Emitter follower.

The input resistance of an emitter follower is defined as the ratio of input voltage V_{in} to base current I_b, i.e.,

$$R_{in} = \frac{V_{in}}{I_b}, \tag{20}$$

but

$$V_{in} = V_{be} - I_e \cdot R = -V_{eb} + \frac{I_b \cdot R}{1-\alpha} \tag{21}$$

and

$$R_{in} = \frac{-V_{eb}}{-I_e(1-\alpha)} + \frac{R}{1-\alpha} = \frac{R_{inb}+R}{1-\alpha}, \tag{22}$$

where $R_{inb} = \frac{V_{be}}{I_e}$ is the input resistance of a grounded-base transistor.

Since usually $R_{in\,b} \ll R$,

$$R_{in} = \frac{R}{1-\alpha},$$

that is, the input resistance of the emitter follower depends on the value of resistor R in the emitter circuit and may be considerably larger than that in a circuit with grounded base or grounded emitter. The voltage gain K_V of the emitter follower is

$$K_V = \frac{V_R}{V_{in}} = \frac{I_e \cdot R}{I_{in} \cdot R_{in}} = K_I \cdot \frac{R}{R_{in}} = \frac{1}{1-\alpha} \cdot \frac{R(1-\alpha)}{R_{in\,b}+R} = \frac{R}{R_{in\,b}+R} \approx 1, \quad (23)$$

that is, the gain of an emitter follower is approximately equal to but always less than unity.

The emitter follower (Fig. 85a) has a bias voltage of +6 v in the emitter-base circuit. This voltage permits increasing emitter current I_e in the absence of an input signal. A voltage of −12 v applied to divider R_1 and R_2 permits obtaining at the transistor base a sufficiently negative voltage to maintain it in the conducting state, whereas current I_e produces at resistor R a voltage drop $I_e \cdot R$ to fully compensate the voltage of +6 v so that the output voltage is zero in the absence of signal. If a negative voltage pulse is applied to the input, the base voltage will become even more negative and will cause an increase in current I_e and an increase in the voltage drop at resistor R. Hence the output voltage, like the input voltage (approximately equal to the input voltage, i.e., −6 v) will be negative. The collector voltage (−7.5 v) is chosen more negative than the largest possible negative input potential (−6 v). This prevents penetration of the collector's charge carriers into the base region and increases the resistance of the emitter-base junction to load the input signal source. In addition, this reduces the input signal rise time. The voltage divider (consisting of resistors R_1 and R_2) and the voltage applied to resistor R_1 (−12 v) are so chosen that the initial input and output signal levels are equal (i.e., equal to zero). As in the case of an inverter amplifier, connection of capacitor C in parallel with resistor R_2 accelerates the output voltage rise and, in addition, decreases the output signal "overshoot" which results from capacitive loading. The emitter-follower output resistance is extremely low; for typical transistors it is 10-50 ohms, which is a substantial advantage of the circuit in its use as a power amplifier.

An emitter follower may use an n-p-n transistor (Fig. 85b). Here, as in the preceding circuit, the voltage between emitter and collector is 13.5 v but is so distributed that −12 v are applied as the emitter bias and +6 v as the base bias. In the absence of an input signal, the emitter current $I_e = I_c + I_b$ produces at resistor R a voltage drop $I_e R$ of 12 v so that V_{out} is equal to zero. In addition, the transistor is completely saturated, since in the base-emitter circuit the voltage is positive. If a negative input voltage is applied, emitter current I_e decreases and there is a simultaneous decrease

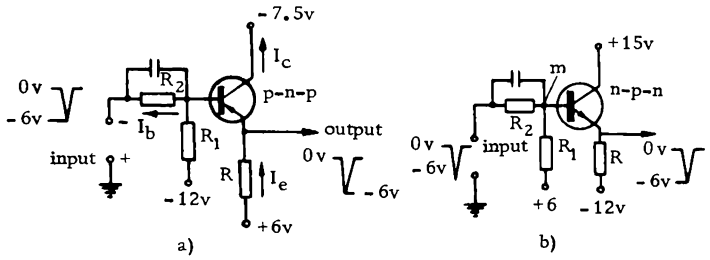


Fig. 85. Emitter followers:
a—with p-n-p transistor; b—with n-p-n transistor.

in the voltage drop across resistor R, with the result that the output voltage, like the input voltage, will be negative (-6 v). In order that the voltage drop at the emitter-base diode does not affect the output voltage, divider $R_1 R_2$ adjusts the voltage at point m to compensate this drop so that the output voltage is equal to the input voltage (Fig. 86a, b).

By parallel connection of several emitter followers we may form a logical AND circuit (Fig. 87) as well as a logical OR circuit (Fig. 88). The output voltage in the circuit of Fig. 88 is positive only in the case where all transistors are cut off by positive voltage at all three inputs. The emitter follower may be used for control of several inverter amplifiers (Fig. 89). Due to the low emitter output resistance, the current I_{on} in the base of transistor T_2 rises rapidly since the emitter potential (point m) becomes negative and a current pulse passes through capacitor C. To cut off transistor T_2, T_1 is near cutoff and the potential at point m becomes close to zero. Current I_{off} flows through resistor R and produces at the base of T_2 a voltage cutting off the transistor. Emitter followers and inverter amplifiers are also used for control of diode logic circuits (e.g., the AND circuit in Fig. 90). The ratio of R_a and R_b is so chosen that the emitter follower conducts current when a negative voltage of -6 v is applied to the input. In this case the

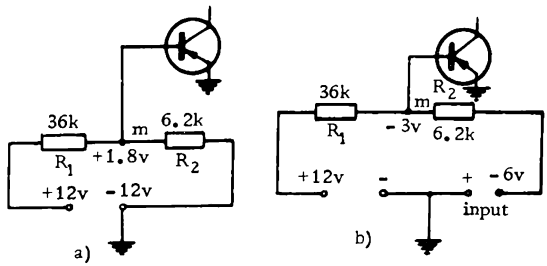


Fig. 86. Base-emitter voltages and currents of emitter follower.

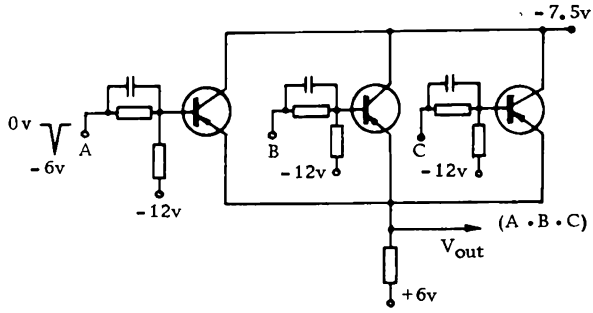


Fig. 87. AND circuit with emitter followers.

output voltage of the emitter follower is negative and diode D_1 conducts. With the application of a positive voltage pulse to the emitter-follower input (from −6 to 0 v), diode D_1 is cut off by a positive pulse at the emitter-follower output and, upon coincidence of the positive pulses cutting off both diodes D_1 and D_2, the V_{out} level of the AND circuit is raised. With an input voltage of −6 v, the inverter amplifier is cut off and has an output voltage close to zero. Hence diode D_2 conducts current through diode D_3. With a positive voltage pulse at the input, the inverter amplifier is in the conducting state and a voltage of approximately −6 v appears at its output. Hence diode D_3 is cut off and diode D_2 does not conduct through it. In short, with the coincidence of proper pulses at the emitter-follower input and the inverter-amplifier input, there is an increase in the voltage level at the output of the diode AND circuit.

The combination of an emitter follower and inverter amplifier permits construction of two-stage power amplifiers with high current gain and high input resistance. In the power amplifier of Fig. 91, transistors T_1 and T_2 have a common resistor in the collector circuit in which the collector currents of both transistors are summed, the result being the control of a relatively large current with a short pulse rise time. With zero input voltage both transistors are cut off. Transistor T_1 is cut off by a positive base potential produced by the voltage divider (resistors of 12 and 3 kilohms) fed from a +12-v source. Transistor T_2 is cut off by a

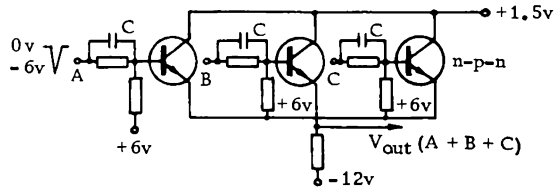


Fig. 88. OR circuit with emitter followers.

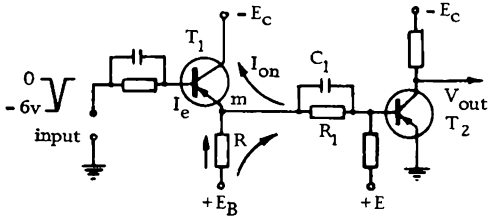


Fig. 89. Inverter amplifier controlled by emitter follower output.

positive base potential from a voltage divider consisting of resistors of 3.9 kilohms and 680 ohms. If a voltage of −6 v is applied to the input of the circuit, the base potential of T_1 becomes negative and the transistor conducts. The voltage drop occurring at the 3.9-kilohm resistor upon passage of the T_1 collector current produces at the base of T_2 a negative potential relative to the grounded emitter; thus T_2 also conducts and the voltage at the output is close to zero. Diode D limits the collector potential of the cut-off transistor to −6 v, with the result that fall and rise of the output pulse occur more rapidly. In the power amplifier of Fig. 92 the first stage is usually an inverter for voltage and current amplification. This circuit is particularly suitable for operation into a capacitive load. With a negative input voltage, transistor T_1 is conducting and its collector current (flowing from emitter to collector) through diode D_1 charges the capacitive load C_L to the potential of the grounded emitter and transistor T_2 is cut off. In the absence of input voltage transistor T_1 is cut off by the positive base voltage from the voltage divider consisting of resistors of 62 and 9.1 kilohms. Transistor T_2 will then be conducting and capacitor C_L charged to a potential of −6 v by the T_2 collector current.

In Fig. 93a the power amplifier with cascaded transistors T_1 and T_2 has two inputs and one output. It is used as an amplifier at the output of a flipflop which in the presence of code 1 turns on transistor T_1 and in the presence of code 0 turns on transistor T_2. In the presence of code 1 the voltage at input 1 is −6 v and at input 2 is

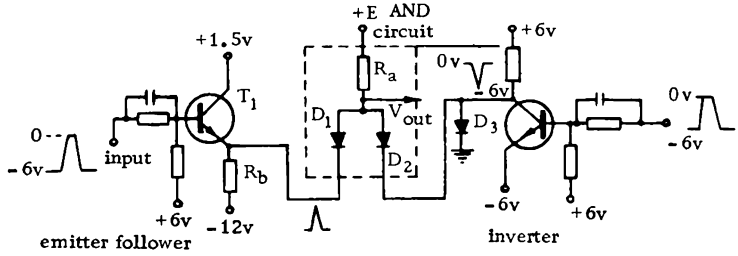


Fig. 90. Control of diode AND circuit from an emitter follower and an inverter.

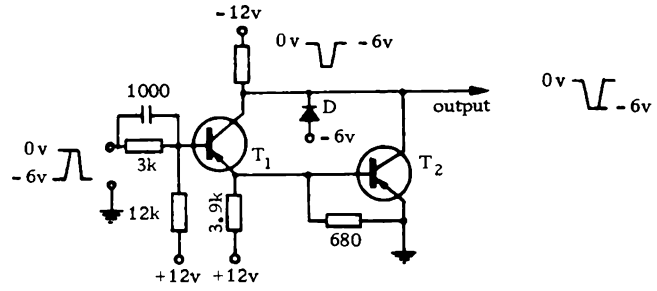


Fig. 91. Power amplifier.

close to zero. Conversely, in the presence of code 0 the voltage at input 1 is zero and at input 2 is −6 v. Transistor T_1 operates as an inverter in the collector circuit of which transistor T_2 is connected as an emitter follower. In the absence of voltage at input 1 (code 0) transistor T_1 is cut off by positive voltage from the divider consisting of R_1 and R_2. At this time, transistor T_2 is conducting, since voltages of −6 v and 0 v are at its input and output, respectively. When the trigger is in the "1" state transistor T_1 is completely conducting and transistor T_2 is cut off; the output voltage increases from −6 to 0 v. Another version of the cascade connection of two transistors is the circuit in Fig. 93b, wherein T_1 is a p-n-p transistor and T_2 is an n-p-n transistor. They constitute a push-pull power amplifier in which, for any polarity of input pulse, the output signal is symmetrical. When the input potential is zero the n-p-n transistor is conducting, with the result that the negative voltage at the emitter (-6 v) is more negative than the potential obtained from the voltage divider consisting of resistors of 33 and 9.1 kilohms. At this time the p-n-p transistor is cut off, since its base obtains a positive voltage from the divider consisting of resistors of 62 and 9.1 kilohms. The resulting output voltage is −6 v, since the n-p-n transistor is saturated. Upon the application of a negative input pulse, the p-n-p transistor is driven to saturation. The n-p-n transistor is cut off because of a sufficiently negative base potential. Since the p-n-p transistor is saturated, the output voltage increases from

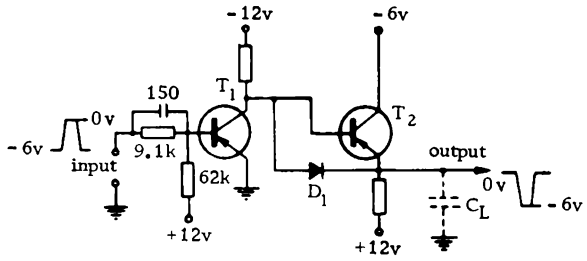


Fig. 92. Power amplifier with capacitive load.

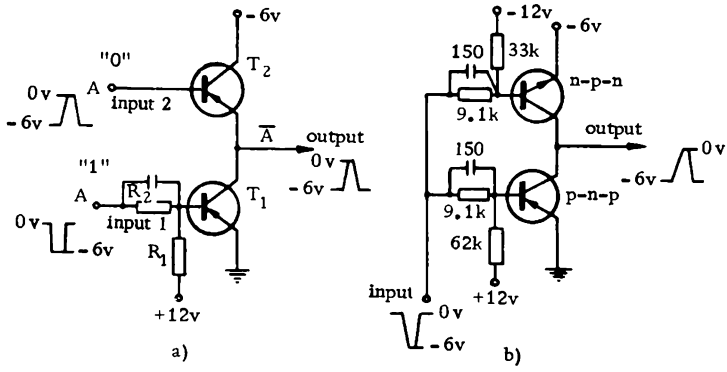


Fig. 93. Power amplifier circuits:
a—with two inputs; b—with common input.

-6 to 0 v and may be directly applied to the load circuit without a load resistor. Due to the use of two complementary transistors, the switching time in this circuit is considerably shorter than in the usual circuit having a saturated inverter. In effect, when the *p-n-p* transistor is cut off, the *n-p-n* transistor is conducting and through it there occurs rapid "resorption" of the current carriers of the *p-n-p* transistor.

4. STATIC TRANSISTOR FLIPFLOPS

Cross coupling of two inverter amplifiers forms a symmetrical static flipflop (Fig. 94a). When transistor T_1 is in the cutoff state (due to application of voltage from the divider consisting of resistors R_1 and R_2), a voltage of -6 v is applied to the input of inverter transistor T_2 through resistor R_{c1} and transistor T_2 is set to the conducting state. With transistor T_2 in the conducting state its collector voltage is close to zero and almost the entire -6 v is applied to R_{c2}. Hence, through the cross coupling, transistor T_1 remains in the cutoff state.

Flipflop switching from one stable state to another (Fig. 94b) is achieved by applying to input 1 or input 2 a negative pulse driving the cut-off transistor to conduct. The pulse is applied to the transistor base through diode D and capacitor C. Resistor R with a large value cuts off one of the diodes D by the negative voltage from the collector of the cut-off transistor. For example, if T_1 is conducting and T_2 is cut off, a negative voltage of -6 v is applied through resistor R to diode D_1 and cuts it off. Diode D_2 is conducting, since the collector voltage of conducting transistor T_1 is close to zero. Hence a negative pulse applied to the common input of the flipflop causes cut-off transistor T_2 to conduct and switches the flipflop to the other stable state. The dashed lines in Fig. 94 show the

connection of both inputs of the flipflop to form a common input. Another method of triggering the flipflop and forming a common input is shown in Fig. 94c. Here the conducting transistor is cut off by a positive pulse applied (to the common input) through capacitor C and diodes D_1 and D_2 to the transistor base. For example, if T_1 is conducting, the positive pulse cuts it off and a negative voltage of -6 v is obtained from its collector. This negative voltage is applied to the base of T_2, causing it to conduct. Hence the positive pulse acts only on the base of conducting transistor T_1, cuts it off, and switches the flipflop to the other state. The operating speed of the flipflop is somewhat lower than the switching speed of two series-connected inverters. This is largely determined by the minority carrier recombination time upon turn-off of the transistor and the recovery time of diodes D_1 and D_2. The pulse rise and fall times may be decreased by inserting an inductance into the collector circuit and limiting the collector voltage by means of clamping diodes fixing the reference voltage level.

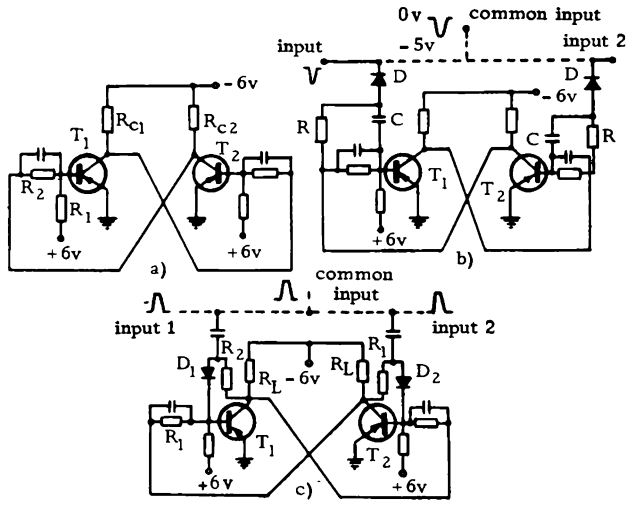


Fig. 94. Flipflop circuits:
a—two cross-coupled inverters; b—flipflop with common input;
c—variant of flipflop with common input.

Figure 95 shows a flipflop circuit with inductances; it is capable of switching at a frequency of the order of 1 Mc. In order to increase the operating speed, inductances of 100 μh are used in the collector circuits; these inductances increase the collector impedance and partially compensate the frequency response of the flipflop. The increase in collector impedance permits increasing the transistor voltage gain K_V, since

$$K_V = -\alpha \frac{Z}{R_{\text{in b}}},$$

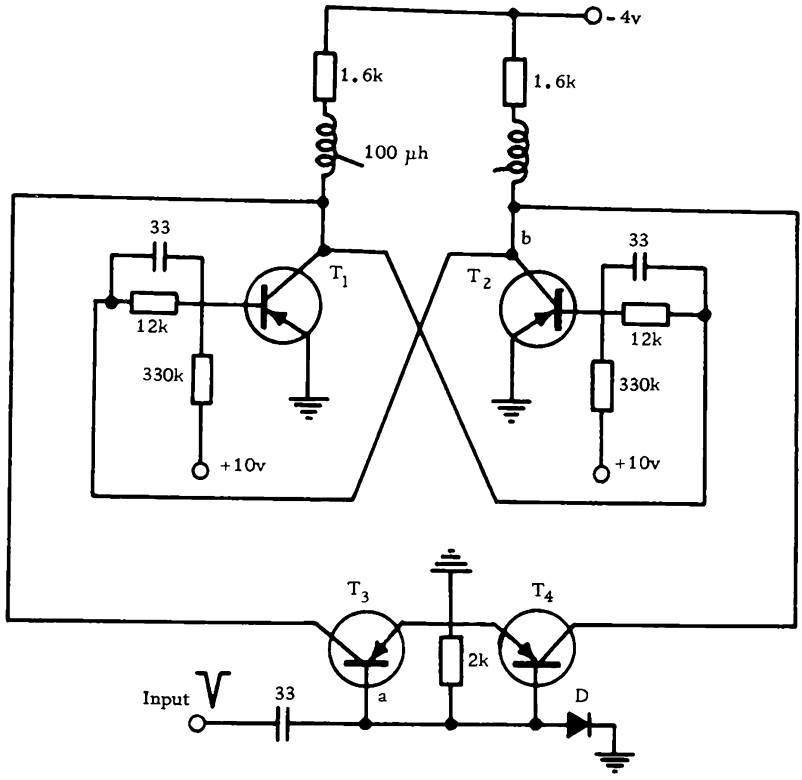


Fig. 95. Flipflop with inductance in collector circuit.

where Z is the impedance in the collector circuit; $R_{in\,b}$ is the input resistance of the common-base transistor. The increase in gain with frequency partially compensates the frequency dependence of the transistor gain. A bias voltage of +10 v reliably cuts off the transistor if the voltage at its input is close to zero. The conducting state of the transistor in this circuit corresponds to a collector voltage of 0.03 v and the cut-off state to a collector voltage of -4 v. When one of the transistors is cut off, a negative voltage of -4 v is applied to the input of the second (conducting) transistor; this voltage is more negative than the bias voltage applied from the voltage divider (330 and 12 kilohms) to the transistor base, and the transistor is held in the conducting state. The flipflop is triggered by means of a balanced circuit consisting of two transistors T_3 and T_4, which replace the usual diodes. The collector voltage is applied to T_3 or T_4. With the application of a negative input pulse, T_3 or T_4 conducts and the voltage at the collector of the cut-off transistor drops to zero. Through the cross-coupling, this voltage is applied to the input of the conducting transistor, cutting it off and changing the state of

the flipflop. Diode D clamps the voltage overshoot due to the presence of the inductances and capacitances. It is seen from the waveforms in Fig. 96 that the input pulses at the bases of transistors T_3 and T_4 are differentiated by means of a capacitor and resistor. The voltage drop at the collector is approximately -4 v and the width of the pulses is approximately twice that of the input pulses, while their repetition frequency is approximately 1 Mc.

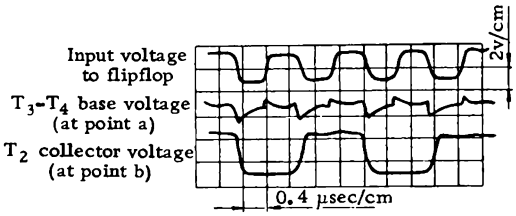


Fig. 96. Flipflop waveforms.

Faster switching of the flipflop may be achieved by replacing the compensating inductances in the collector circuit with a ferrite-core transformer (Fig. 97a). In this case, in addition to the base feedback from the cross-coupling, there is transformer coupling between the collector circuits of the two transistors. The transformer windings are connected in opposite phase. Hence, a negative current pulse in the circuit of a cut-off transistor, for example, in T_1, induces a positive pulse in the other winding of the transformer. This additional pulse is transmitted through the base cross-coupling to the base of transistor T_1 and causes its cutoff.

When transistor T_2 conducts a positive pulse occurs in the winding of transformer TR, inducing in the secondary winding a negative pulse which is fed back to the base of T_2 and causes it to conduct. The result is a more rapid switching of conducting transistor T_2. Moreover, the current pulse in the collector circuit of the cut-off transistor T_1 accelerates charging of the capacitor and decreases the recombination time of minority carriers in the base of T_1. Thus, the use of transformer feedback during switching of the flipflop shortens the pulse rise and fall time. For example, for a transistor flipflop circuit with transformer coupling the duration of the triggering was 1.5-2 times less than for the same flipflop without transformer coupling.

Transformer coupling is also used in the circuit in Fig. 97b. When T_1 is conducting and T_2 is cut off, the collector potential of T_1 is 0.1 v more negative than its emitter potential (due to the base-emitter current). Since this negative voltage produces a drop at the 220-ohm resistor, the base of T_2 has a positive potential relative to its emitter and T_2 is cut off. On the other hand, if T_1 is cut off, a negative voltage from the divider consisting of R_1, R_2, R_3 is applied to the base of T_2 and, since R_1 is greater than $R_2 + R_3$, the negative base voltage is limited by diode D_1 to 0.5 v (due to the voltage drop at the diode itself). When a negative pulse is

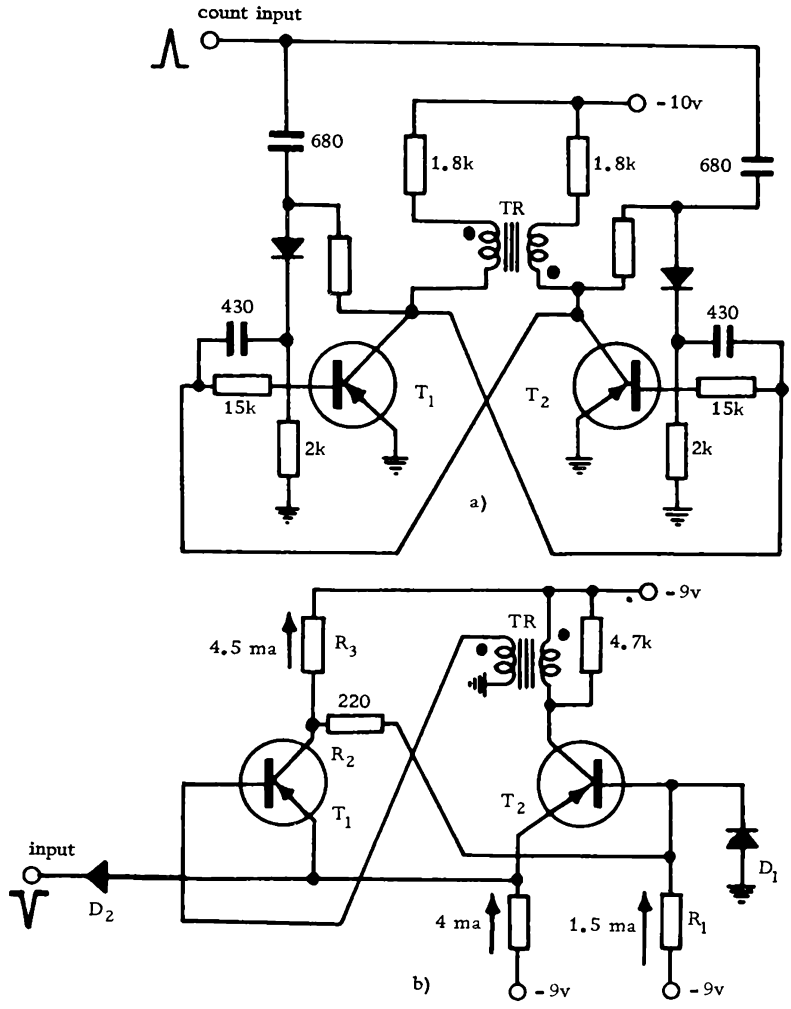


Fig. 97. Transformer-coupled flipflops.

applied to the input through diode D_2 the conducting transistor T_2 is cut off by this pulse and the current in the collector circuit of T_2 stops. In addition, a relatively large voltage overshoot occurs in the secondary winding of transformer TR. The turns ratio is 4:1. Rapid damping of the transient occurring upon removal of current in the transformer primary is achieved by a 4.7-kilohm shunt resistor. In order to ensure reliable operation of the circuit with a possible 30-percent spread in the values of transistor parameters, the required peak overshoot with the chosen turns ratio was 4.2 v. The overshoot voltage rapidly "resorbs" the accumulating

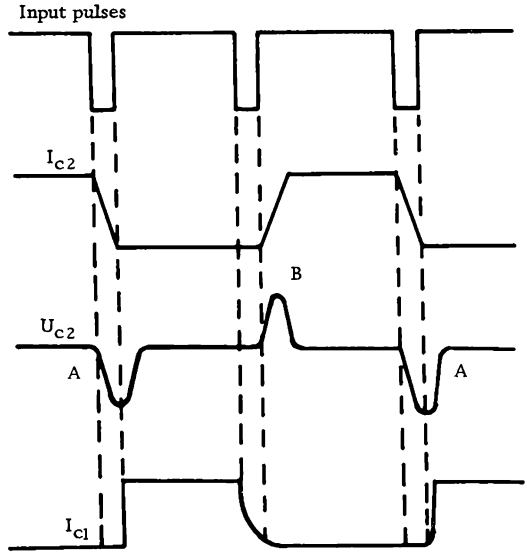


Fig. 98. Waveforms in flipflop of Fig. 97b.

minority carriers at the base of transistor T_1 and the retarding action of the accumulated carriers has little influence on the switching speed. The current and voltage waveforms of this circuit are shown in Fig. 98. The turn-off delay of the transistor was 1 μsec and the turn-on delay was 0.5 μsec. Thus, a total time of 1.5 μsec is required to switch the flipflop. In order to increase the operating

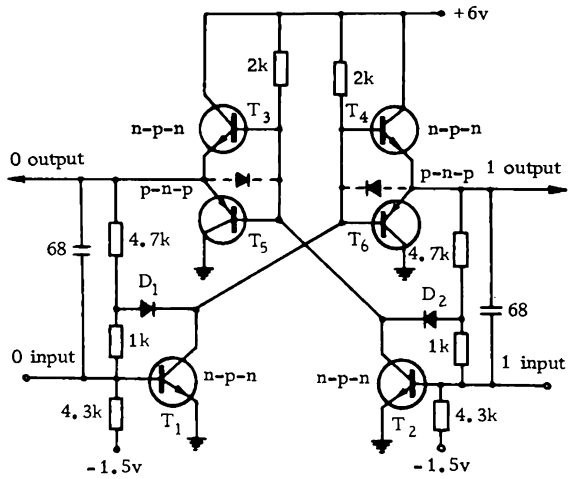


Fig. 99. Flipflop with emitter follower at output.

speed of the flipflop more complex circuits are used. Emitter followers are connected as power amplifiers (Fig. 99).

In Fig. 99 transistors T_1 and T_2 (n-p-n) are switched by input pulses applied directly to their bases. In the absence of input signals one of the transistors is cut off by a bias voltage of -1.5 v and the other is conducting. At each transistor output there is connected an emitter follower consisting of two series-connected transistors T_3, T_5 and T_4, T_6, wherein T_5 and T_6 serve as resistors in the emitter circuits of transistors T_3 and T_4. If T_1 is conducting and T_2 is cut off, transistor T_4 is almost cut off, since the entire voltage $E_c = +6$ v is applied to the 2-kilohm collector resistor. Since T_2 is cut off and the 5-v collector supply is applied to the base of T_3, T_3 is completely conducting. A voltage of 0.45 v is maintained at the collector of the conducting transistor T_1 by means of the divider consisting of resistances of 4.7, 1 and 4.3 kilohms and diode D_1.* This voltage ensures operation of the conducting transistor T_1 at the saturation threshold and eliminates the effect of accumulation of holes (minority carriers), thus decreasing the turn-off time.

As intermediate amplifiers, emitter followers T_3 and T_4 at the outputs of T_1 and T_2 discharge the latter from the load of the feedback loop (consisting of C = 68 $\mu\mu$fd and a combined 5.7-kilohm resistance) as well as from the external load at the flipflop output. The result is rapid switching of the flipflop, and the amplitude of the output voltage pulse is approximately 6 v.

The stable state of the flipflop is prolonged due to the feedback loop consisting of resistances of 4.7 + 1 kilohms connected in the emitter circuit of the conducting transistor T_3; hence T_1 remains conducting after disappearance of the input pulse. Transistor T_2 remains cut off since a bias voltage of -1.5 v is applied to its base. The voltage at output 1 is close to zero since transistor T_4 is cut off and transistor T_6 is conducting. The 68-$\mu\mu$fd capacitor shunts the 5.7-kilohm resistor, accelerating the flipflop switching. The load current is 5 ma; the turn-on time is approximately 0.3 and the fall time 0.2 μsec. The unloaded flipflop draws 25 mw. The circuit operates at frequencies up to 2 Mc over a temperature range from -55° to 85°C (using transistors of type 2N94A).

In order to increase the operating speed of the flipflop, circuits are isolated from the trigger pulse sources and output circuits from the load by means of intermediate amplifiers. Fig. 100 shows the diagram of a high-speed flipflop in which the input transistor amplifiers T_3 and T_4 are connected in series with transistors T_1 and T_2 of the flipflop proper. In the absence of pulses at the input, transistors T_3 and T_4 are conducting. If a positive pulse is applied to input 1, transistor T_3, being cut off, interrupts the current flow of T_1 of the flipflop. The previously cut-off transistor T_2 now conducts since a voltage of 3 v is applied from the collector

---

* $\frac{+6\text{ v} - 1.5\text{ v}}{4.7\text{ k} + 1\text{ k} + 4.3\text{ k}} = \frac{4.5\text{ v}}{10\text{ k}} = 0.45$ v per 1 kilohm.

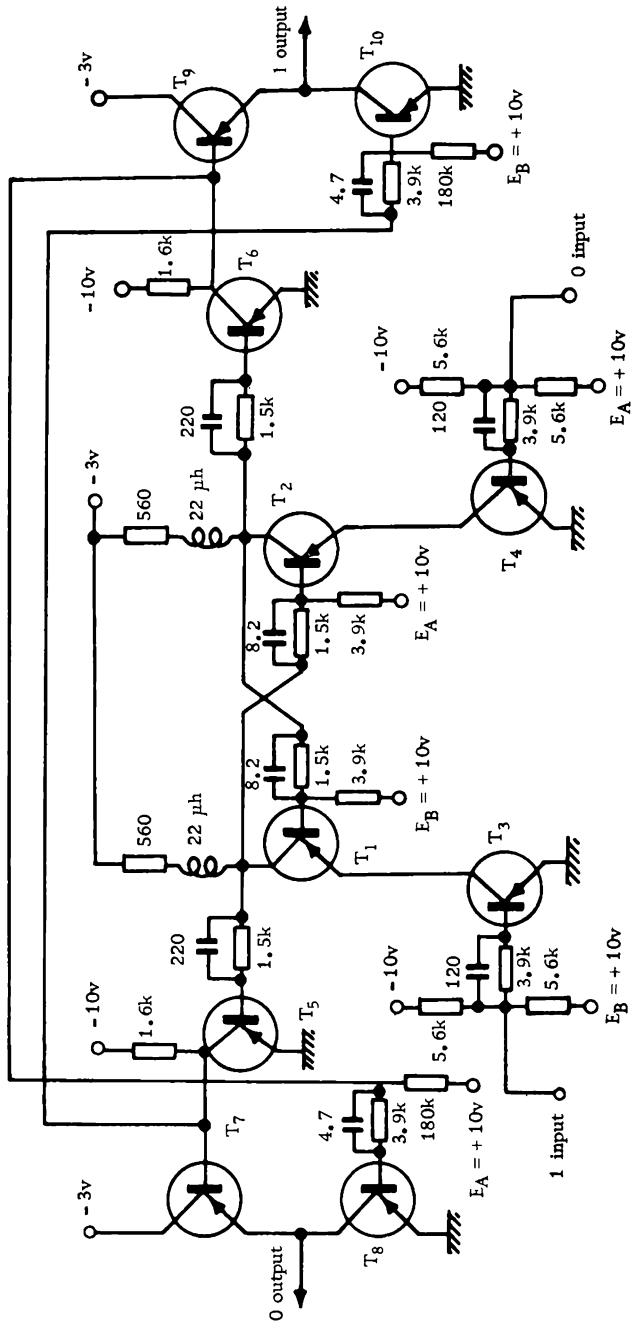


Fig. 100. High-frequency flipflop with emitter followers at input and output.

of T_1 through the feedback loop. Transistor T_1 will not conduct after disappearance of the input pulse since a positive voltage of approximately 0.4 v* will be applied to its base from the voltage divider consisting of resistors of 39 and 1.5 kilohms. At the outputs of transistors T_1 and T_2 there are connected inverter amplifiers T_5 and T_6. With T_1 in the cutoff state, T_5 will be conducting since a voltage of 3 v is applied to its base through resistors of 560 ohms and 1.5 kilohms. Transistor T_6 is then cut off since T_2 conducts current and the voltage at the base of T_6 is close to zero. Inverter amplifiers T_5 and T_6 control emitter followers T_7 and T_9 with T_8 and T_{10} in their emitter circuits.

With T_5 conducting and T_6 cut off, T_7 will be cut off since the potential at its base is close to zero. Transistor T_9 will then be conducting since T_6 is cut off and to the base of T_9 there is applied (through the 1.6-kilohm resistor) a negative voltage of -10 v. Transistor T_{10} is cut off because the output voltage of conducting T_5 is close to zero and a positive voltage through the divider consisting of 180- and 3.9-kilohm resistors is applied to the base of T_5. Thus, with T_1 of the flipflop in the cutoff state there is a voltage of -3 v at the 1 output and a voltage close to zero at the 0 output since, due to cross coupling between the input of transistor T_8 and the output of T_6, on the one hand, and the input of T_{10} and the output of T_7, on the other hand, transistor T_8 will be conducting and T_{10} will be cut off. The presence of this additional cross coupling also facilitates rapid switching of the output circuits of the flipflop. Using diffused-junction and "micro-power" transistors, the described flipflop has a switching frequency of 10 Mc. Figure 101 shows the waveforms at the flipflop output with and without load. Sensitivity of the flipflop does not vary with the load and the dependence of pulse amplitude (in v) on frequency has a characteristic parallel to the abscissa axis. Among the advantages of the flipflop is the possibility of high-frequency operation without diodes, which reduces the type of components.

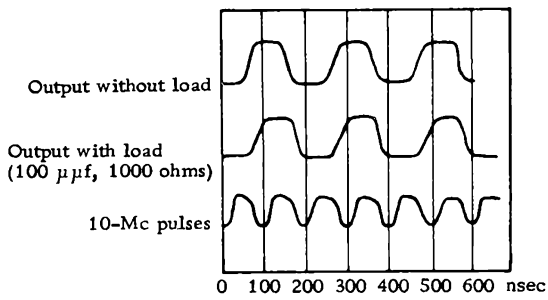


Fig. 101. Waveforms for flipflop of Fig. 100.

*$\frac{10 \text{ v} \cdot 1.5 \text{ k}}{39 \text{ k} + 1.5 \text{ k}} = 0.4$ v per 1.5 kilohms.

5. UNSYMMETRICAL STATIC FLIPFLOPS WITH EMITTER COUPLING

The static Schmitt trigger has emitter feedback (Fig. 102). It is similar to a vacuum-tube flipflop with cathode coupling and has two stable states. Assume that initially transistor T_1 is cut off and T_2 is conducting. Then the current passing through T_2 produces at resistor R_e a voltage drop of $I_{c_2} \cdot R_e$, which completely cuts off transistor T_1. Transistor T_2 is in the conducting state since, from the collector of T_1 through the voltage divider consisting of resistors of 5.6, 50 and 3 kilohms, there is applied to its base a negative voltage of -1.2 v which is more negative than the voltage drop $I_{c_2} \cdot R_e$. If a negative pulse of such a magnitude that the voltage between emitter and base is greater than zero is applied to the base of transistor T_1, T_1 becomes conducting and T_2 becomes cut off. Thus, as soon as T_1 conducts, a collector current I_{c_1} flows through resistor R_e and produces a more negative voltage drop at resistor R_e. At the same time, the potential at point a becomes less negative, with the result that T_2 becomes cut off. This will cause a decrease in the current I_{c_2} flowing through resistor R_e. For transistor T_1, the decrease in current I_{c_2} through resistor R_e is a positive feedback, as is the positive voltage drop at point a and at the base of transistor T_2. But the increase in current I_{c_1} causes an increase in the voltage drop at R_e, which is a negative feedback for transistor T_1. When the positive feedback is considerably greater than the negative feedback, transistor T_1 conducts and transistor T_2 is cut off. In order for the flipflop to return to the initial state a positive pulse must be applied to the base of T_1. The waveforms of the input and output pulses of the flipflop are shown in Fig. 103. The threshold voltage values at which the flipflop is switched are not identical in transition from the 0 state to the 1 state and vice versa. By means of the voltage divider the input voltages V_{10} and V_{01} are so set that they lie in the zone $\Delta = V_{10} - V_{01}$. Input voltages V_{10} and V_{01} are +10.9 and -11 v, respectively. The advantage of this flipflop is the absence of reactive components and its relatively high operating speed.

Another flipflop with emitter coupling is shown in Fig. 104, wherein T_1 operates as an inverter amplifier and T_2 as an emitter

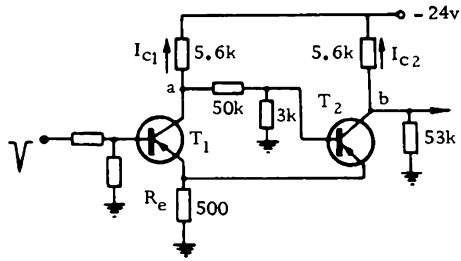


Fig. 102. Schmitt trigger.

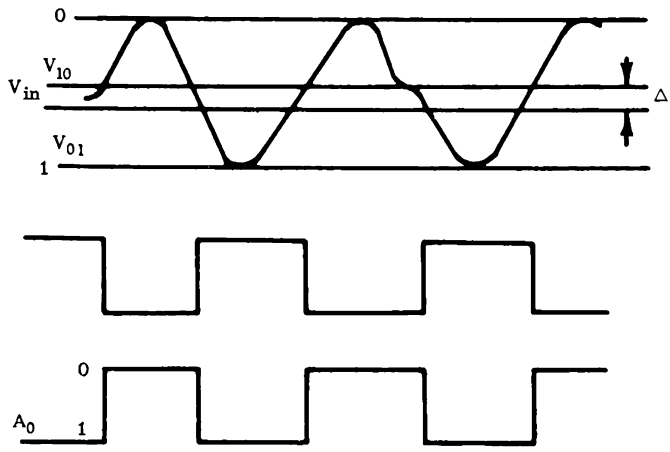


Fig. 103. Waveforms for Schmitt trigger of Fig. 102 (V_{01} = 11 v, V_{10} = 10.9 v).

follower with common emitter resistor of 1.1 kilohms. When T_1 is cut off its collector is held at a negative voltage of -1.5 v by means of limiter diode D_2, which decreases the amplitude of voltage drop at the collector and consequently accelerates the switching process. With T_1 in the cut-off state, transistor T_2 conducts due to the negative voltage of -1.5 v at its base. In order to increase the emitter resistance in the T_2 follower circuit and to decrease the negative feedback in the T_1 circuit, a voltage divider consisting of resistors 1.7 and 1.1 kilohms is used. The +3-v source serves to increase the level of current in the circuit of the conducting

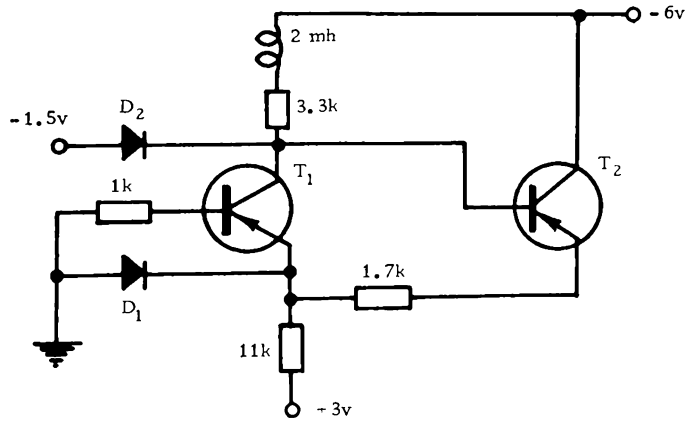


Fig. 104. Flipflop with emitter coupling.

transistor and to decrease the signal amplitude during switching. Diode D_1 improves the switching time of the transistor.

6. STANDARD ELEMENTS OF TRANSISTORIZED DIGITAL COMPUTERS

The inverter amplifier is a standard element for digital computers. Figure 105 shows one in which there are two inputs connected to the base of transistor T_1. In the absence of input signals T_1 is cut off by the positive bias voltage at the base (+12 v), which is twice the positive potential (+6 v) at the emitter. If a negative signal larger than 6 v is applied to input 1 or 2, transistor T_1 will conduct. The 270-$\mu\mu$fd capacitor connected in parallel with resistor R accelerates the switching. When the transistor is cut off, the output voltage is clamped to ground potential by diode D and almost all the voltage of -18 v is across the 1.5-kilohm collector resistor. When T_1 becomes conducting, a larger current flows through the collector resistor and a positive pulse appears at the output with a magnitude of about +6 v (the saturated resistance of T_1 is considered negligible). Two of these standard elements can be used to form a static flipflop. This element can be used as an AND circuit or as an OR circuit; it is thus a dual-purpose element.

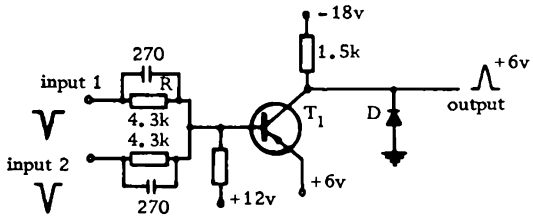


Fig. 105. Inverter amplifier as standard element.

Figure 106 shows the schematic of a delay element. The circuit consists of two transistors T_1 and T_2. In the absence of an input signal, T_1 is cut off by the +12-v bias voltage at its base and T_2 is conducting since its emitter is at a positive voltage of +6 v relative while its base is grounded through resistor R. The collector potentials of T_1 and T_2 are, respectively, close to zero and approximately +6 v. Capacitor C is at +6 v at point n and is clamped to zero at point m. When a negative pulse is applied to the input of T_1, it conducts and point m is at a potential of +6 v. Hence the capacitor is recharged through resistor R and the voltage drop resulting from capacitor recharge cuts off transistor T_2. Transistor T_2 remains cut off until capacitor C is completely discharged through resistor R. The positive voltage pulse at the output of T_2 appears only for a certain time determined by the delay of the RC circuit in recharging of the capacitor.

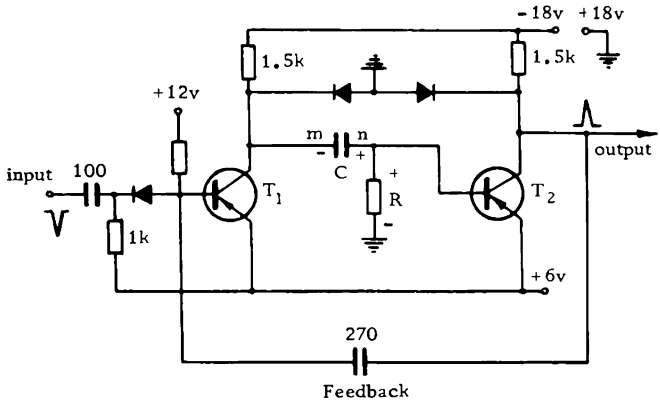


Fig. 106. Standard delay element.

For pulse repetition frequencies of the order of several hundred kilocycles, standard elements based on an inverter amplifier or emitter follower are used. In the absence of input signal, the inverter amplifier in Fig. 107a is in the cut-off state, since both base and emitter are at ground potential. The output voltage is clamped by diode *D* to -2 v. The collector voltage supply produces a current which passes through resistor *R* and diode *D*. When an input pulse of -2 v is applied to the input, the transistor conducts in a saturation condition. The output voltage rises from -2 v to zero.

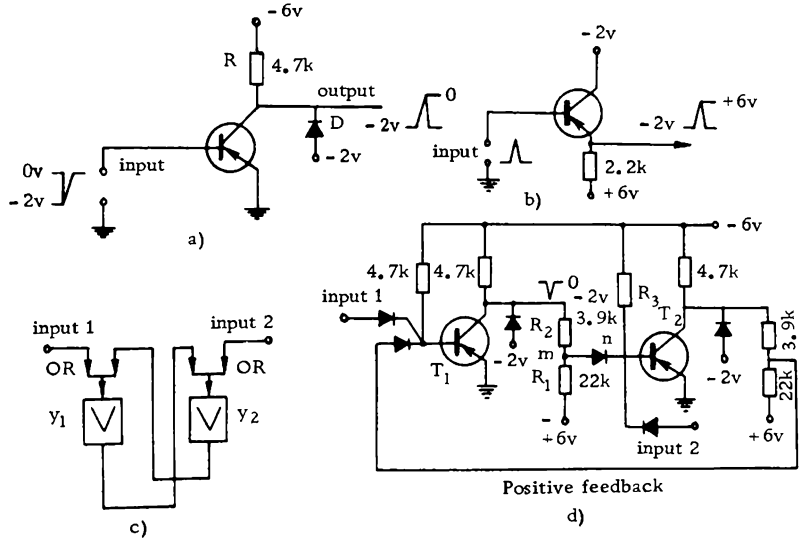


Fig. 107. Standard semiconductor elements:
a—inverter amplifier; b—emitter follower; c—block diagram of flipflop; d—flipflop with two cross-coupled inverters.

In the absence of input signal, the emitter follower in Fig. 107b is normally conducting and a relatively large current flows through resistor R since the emitter voltage is above ground due to emitter supply voltage of +6 v. The output voltage is -2 v. If a positive pulse of 6 v is applied to the input, the transistor becomes cut off.

Two inverter amplifiers form a flipflop circuit (Fig. 107c and d). The flipflop operates as follows. With T_1 conducting, its output voltage is close to zero. The current passing through diode D from the collector voltage supply of -6 v generates at the base a potential of approximately +1 v, hence T_2 will be cut off.

If a positive pulse is now applied to the input, T_1 is cut off and the output becomes -2 v; the voltage at point m of divider R_1 and R_2 is approximately -1 v. The current flowing through resistor R_3 creates a potential of -1 v at the base of T_2 and T_2 becomes conducting. The presence of positive feedback reinforces transition of T_1 from the conducting state to the cut-off state and holds it at the new state. A similar situation occurs when a positive pulse is applied to input 2.

7. HIGH-SPEED CURRENT SWITCHES

The large amplitudes of signals at the input and output of switching circuits have resulted in transistor operation in saturation and the switching frequency has been limited by the minority carrier storage effect.

Below we discuss high-speed current switches having transistors operated under nonsaturated conditions at which low input voltages of about ±0.6 v and practically constant current sources are used. The schematic of a high-speed current switch (Fig. 108) is a differential amplifier with transistors T_1 and T_2. The control signal is applied to the base of T_1, while a fixed potential (ground potential in this case) is applied to the base of T_2. The circuit includes three current sources (1-3) which provide constant currents of 3 and 6 ma regardless of change in external load. The current sources consist of voltage supplies E_A, E_B and E_C connected in series with large resistance R. A supply voltage E_c of -6 v is connected to the collector circuits, each of which includes a 200-ohm resistor and an inductance for increasing collector resistance upon the passage of pulses. The input voltage at the base of the conducting transistor T_1 swings from -0.6 v (conducting) to +0.6 v (cut off). When the input voltage at the base of T_1 is -0.6 v, T_1 conducts a current of 6 ma since almost the entire voltage of supply E_A is applied to the large resistor R. The emitter potential is close to zero when the base is at a negative voltage of -6 v. At this time, transistor T_2 is cut off. The 6-ma current passing through conducting transistor T_1 branches out in the collector circuit into two equal parts, one into current source 2 and the other through collector resistor R_{c1} of 200 ohms, producing a voltage drop of 0.6 v ($=20 \times 3 \times 10^{-3}$). This voltage is subtracted from the collector voltage so that the output voltage level with transistor T_1 in the

conducting state is $-6 + 0.6 = -5.4$ v. The voltage of cut-off transistor T_2 at output B is -6.6 v ($=-6.0-0.6$), since the 3-ma current passing through resistor R_{c_2} produces a voltage drop of 0.6 v. If a voltage of +0.6 v (a positive pulse) is now applied to the base of T_1, this transistor is cut off as the emitter has a potential close to zero; T_2 then becomes in the conducting state as the emitter potential of T_2 will increase relative to its base. T_2 will conduct a current of 6 ma from source 1; the current then branches out, one branch into source 3 (3 ma) and the other through resistor R_{c_2} (200 ohms), producing a voltage drop of +0.6 v. Hence in switching of the transistors the potential at output B is -5.4 v ($= -6.0 + 0.6$) and at output A the potential is -6.6 v ($=-6.0-0.6$). The presence of two outputs A and B gives both a direct signal and an inverted signal, which eliminates further need of inverters at the output.

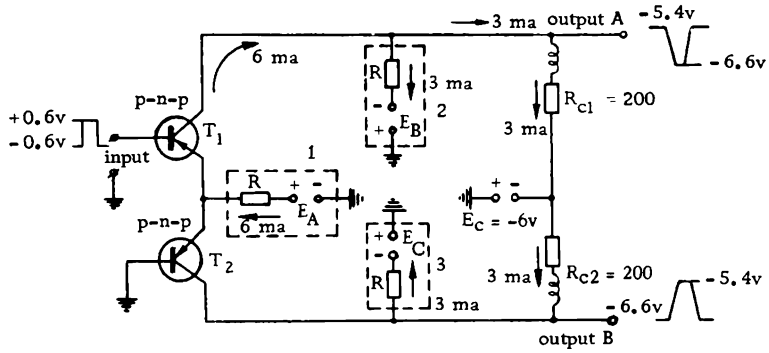


Fig. 108. High-speed current switch using p-n-p transistors.

A disadvantage of the above high-speed switch is the difference in output signal level (6 v) and input signal level (0.6 v), which prohibits direct coupling of one switching circuit to another. This disparity may be eliminated by having an additional high-speed current switch using $n-p-n$ transistors (Fig. 109). A voltage of -6 v is applied to the base of T_2, while the base voltage of T_1 varies from -6.6 to -5.4 v, which corresponds to the output voltages of the above-described circuit with $p-n-p$ transistors. When the base voltage of T_1 is -6.6 v it is cut off because its emitter potential (at point a) is approximately -6 v. Transistor T_2 is in the conducting state since the voltage at its base is higher than that at the emitter. The voltage at output A is equal to a voltage drop of up to +0.6 v at resistor $R_{c_1} = 200$ ohms from the 3-ma source 2. With T_2 in the conducting state the 6-ma current at sink 1 is merged from two equal sources, one from current source 3 and the other from the current flowing through collector resistor R_{c_2} (200 ohms), producing a voltage drop of -0.6 v.

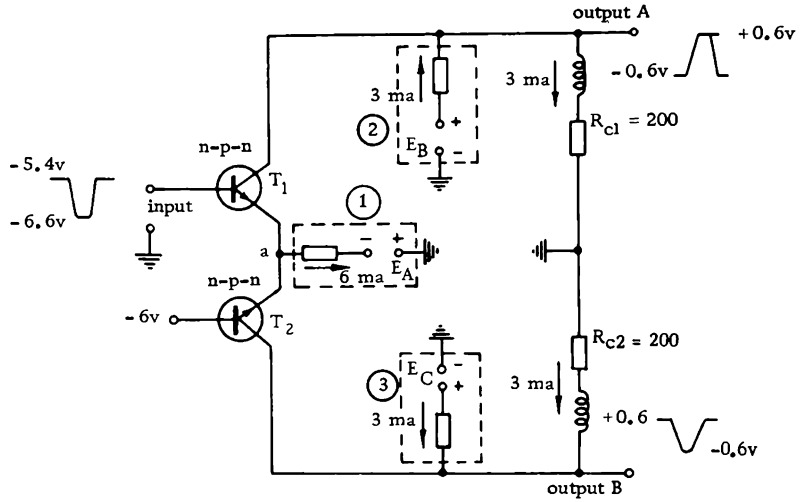


Fig. 109. High-speed current switch using n-p-n transistors.

When a voltage of 5.4 v is applied to the base of T_1, switching of the transistors occurs. Since the base potential of T_1 becomes less negative than that of the emitter (-6 v), transistor T_1 conducts. The emitter potential of T_2 (point *a*) is close to -5.4 v, and is less negative than the base potential (-6 v) of T_2; hence T_2 becomes cut off and the 6-ma current flowing into sink 1 passes through T_1. Three ma of the current from source 1 produce a voltage drop of -0.6 v at resistor R_{c_1} since the direction of the current is the reverse to that which occurred when T_1 was cut off. There is also a change in the direction of current which flows through resistor R_{c_2} since with T_2 in the cut-off state this current flows from source 3. In this case, the voltage at output *B* is 0.6 v.

A single switching circuit using *p-n-p* transistors can be used to control several circuits with *n-p-n* transistors. One of the outputs of the switching circuit with *p-n-p* transistors is connected to the inputs of several *n-p-n* switching circuits connected in parallel (Fig. 110). The low collector resistance of the switching circuit (about 200 ohms) permits controlling a number of switching circuits with *n-p-n* transistors.

By the addition of transistors in parallel with T_1 we may have a number of logic circuits having *n* inputs. Fig. 111 shows the schematic of a circuit for the logical OR operation at one output *A* and NOR operations at the other output *B*. A voltage of +0.6 v at the bases of transistors $T_1 \ldots T_n$ corresponds to code 0 and a negative voltage of -0.6 v to code 1. A negative voltage of -6.6 v at output *A* or *B* corresponds to code 0 and a negative voltage of -5.4 v (−6 + 0.6 v) to code 1. If a positive voltage of 0.6 v is applied to all transistors $T_1 - T_n$, then output *A* has code 0 (-6.6 v) and output *B* code 1 (-5.4 v). Thus, in the absence of signals S_1, $S_2 \ldots S_n$ at the $T_1 \ldots T_n$ inputs (code 0), a voltage corresponding

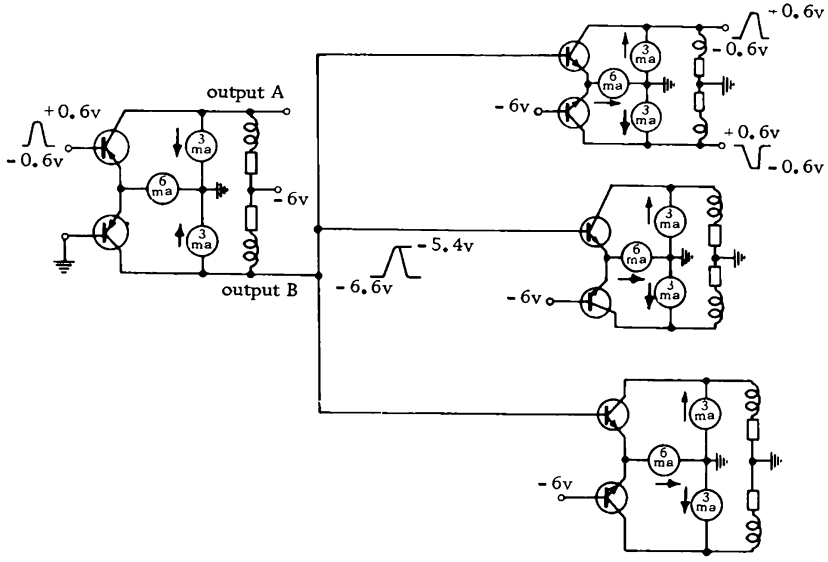


Fig. 110. Control of several switching circuits.

to code 1 is obtained at output B, a logical NOR (NOT—OR) operation. If a voltage of -0.6 v (code 1) is applied to the input of one of the transistors $T_1 - T_n$, output A has code 1 (-5.4 v), a logical OR operation.

Figure 112 shows the waveforms of voltage pulses at the input and output of the OR circuit in Fig. 111, where three transistors $T_1 - T_3$ are connected in parallel and two of the three are in the state corresponding to code 0 (absence of voltage). A pulse voltage (curve 1) at a frequency of 10 Mc is applied to the third input. The voltages at outputs A and B are shown in curves 2 and 3. It is

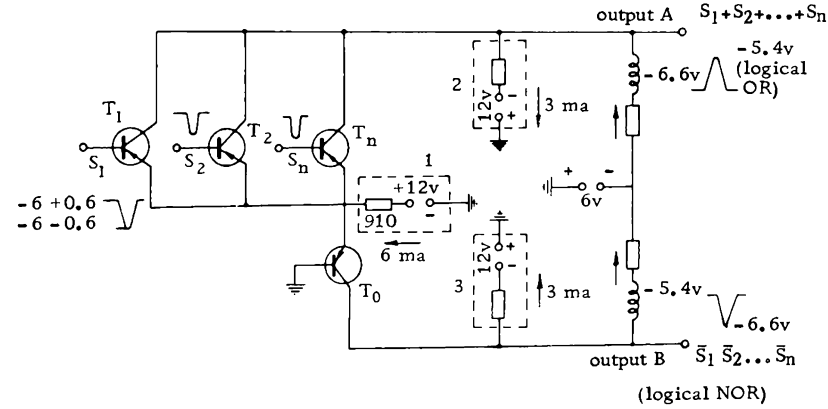


Fig. 111. Circuits for logical OR and NOR operations.

seen from the waveforms that the output pulses are also repeated at a frequency of 10 Mc with a delay of not greater than 20 nsec.

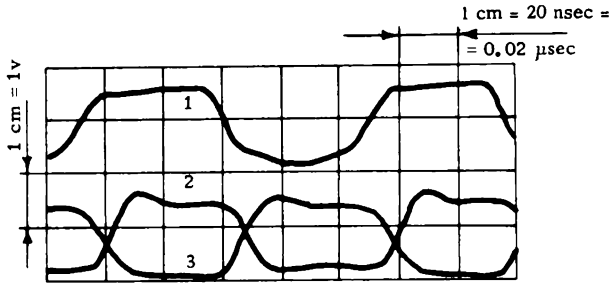


Fig. 112. Waveforms of voltage pulses.

Figure 113 shows the waveform for the same circuit (Fig. 111) wherein the input voltage pulse at a frequency of 10 Mc is applied to the base of the two upper transistors and the third is in the state corresponding to code 0. The voltages applied to the bases of T_1 and T_2 (curves 1 and 2, respectively) are inverted relative to one another and have a repetition frequency of 10 Mc. Curve 4 shows the voltage at output A (OR), and curve 3 the voltage at output B (NOR). The NOR circuit output signal is shown almost at a constant level while that of the OR circuit has a small value and ideally should be constant.

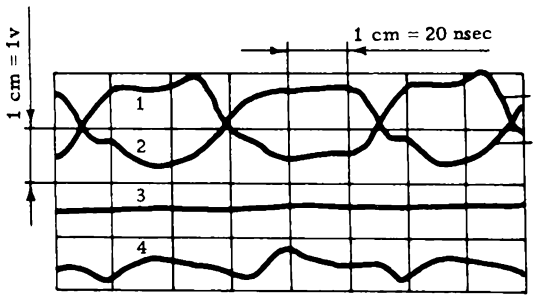


Fig. 113. Waveforms of voltage pulses when third upper transistor is in the state corresponding to code 0.

The exclusive-OR circuit (Fig. 114) consisting of six transistors is used as an adder for generation of the sum of two quantities A and B. Let us assume that a voltage of -0.6 v at the base of the upper transistor corresponds to code 1 (conducting) and a voltage of +0.6 v corresponds to code 0 (cut-off). There are four possible combinations of codes A and B:

a) $A = 0$; $B = 0$ ($\bar{A} = 1$, $\bar{B} = 1$*). In this case the lower transistors T_1 and T_4 are cut off as are T_2 and T_6, whereas T_3 and T_5 are conducting. Since a current of 6 ma passes through the upper conducting transistors T_3 and T_5, the voltage at output 2 is -6.6 v, which corresponds to code 0 while the voltage at output 1 corresponds to code 1 (-5.4 v).

b) $A = 1$, $B = 0$ ($\bar{A} = 0$, $\bar{B} = 1$). Transistors T_2 and T_3 are conducting and T_5 and T_6 are cut off. A current of 6 ma passes through the conducting transistor T_3 and produces at resistor R_{C2} a voltage drop of +0.6 v. Hence the voltage at output 2 corresponds to code 1 (-5.4 v) and at output 1 to code 0.

c) $B = 1$; $A = 0$ ($\bar{B} = 0$, $\bar{A} = 1$). Transistors T_5, T_6 and T_1 are conducting and T_2, T_3 and T_4 are cut off. A current of 6 ma produces the same voltage drop at resistor R_{C2} as in the previous case. Hence the voltage at output 2 corresponds to code 1 and at output 1 corresponds to code 0.

d) $A = 1$; $B = 1$ ($\bar{A} = 0$, $\bar{B} = 0$). In this case T_1 and T_4 are cut off and T_2 and T_6 are conducting. The voltage at output 2 corresponds to code 0 and at output 1 corresponds to code 1.

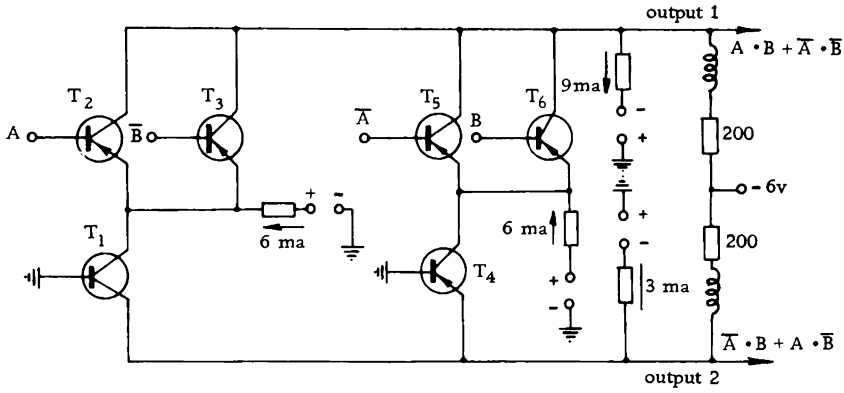


Fig. 114. Exclusive-OR circuit.

Thus, one output of this circuit gives the logical exclusive-OR operation $\bar{A} \cdot B + A \cdot \bar{B}$, and the other gives the logical coincidence operation $\bar{A} \cdot \bar{B} + A \cdot B$.

The 9-ma current source is chosen in order to ensure passage of a current of -3 ma through resistor R_{C1} (with T_1 and T_4 cut off) and a current of +3 ma in the remaining cases. In effect, when T_1 and T_4 are cut off, the current of 12 ma from the two 6-ma sources is split so that part of it (9 ma) passes through the 9-ma source and the remaining 3 ma flows through resistor R_{C1}, producing therein a positive voltage drop of +0.6 v.

*The logical NOT operation (negation) is denoted by a bar above the letter ($\bar{A}$, i.e., NOT A).

Similarly, by connecting a third transistor in parallel with the two upper transistors we may obtain a circuit (Fig. 115) for logical operations of three quantities A, B and C. Output 2 provides the logical operation: $A \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C$ and output 1 provides the complement of output 2 (i.e., $\overline{A \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C}$).

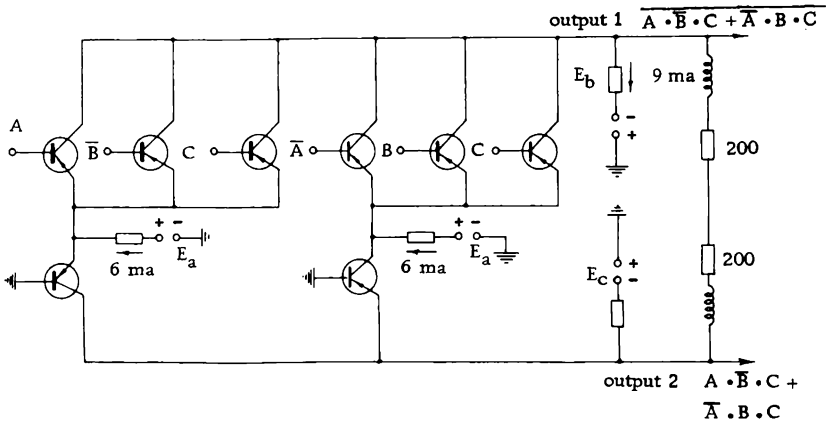


Fig. 115. Circuit for operations with three quantities.

The use of transistor current switches as computer elements, although insufficiently investigated, offers prospects of operation of elements at extremely high frequencies since drift transistors can be made for operation at frequencies up to 500 Mc. These transistors permit raising the collector breakdown voltage to 50 v and higher. Another advantage is the low output resistance and the almost complete absence of noise. A disadvantage is high collector dissipation at relatively large collector voltages.

Logic circuits similar to those described above may be constructed with the use of n-p-n transistors (Fig. 116). A voltage of -5.4 v at the input of one of the transistors T_1—T_n corresponds to code 1 and that of -6.6 v corresponds to code 0. When all transistors T_1—T_n are cut off (code 0) T_0 is conducting and the voltage at output A is -0.6 v, which corresponds to code 1. Thus, the NOR operation is executed (i.e., NOT-OR). If a positive voltage pulse of -5.4 v is applied to only one of the inputs, the corresponding transistor becomes conducting and T_0 is cut off; the voltage at output B corresponds to code 1, i.e., the logical OR operation.

The cross coupling of two current switches using p-n-p and n-p-n transistors forms a flipflop (Fig. 117) with two stable states. T_1 and T_2 form the first half of the switching network and T_3 and T_4 form the second half. In one state, T_1 is cut off by a voltage of +0.6 v from the output of T_4 and transistor T_2 is conducting, while T_3 is conducting due to the voltage from the output of T_2 (-5.4 v) and T_4 is cut off. If a negative pulse is applied to the base of T_5, which is connected in parallel with T_1, T_5 conducts and T_2 becomes cut off. Due to the cross coupling, a voltage of -6.6 v is

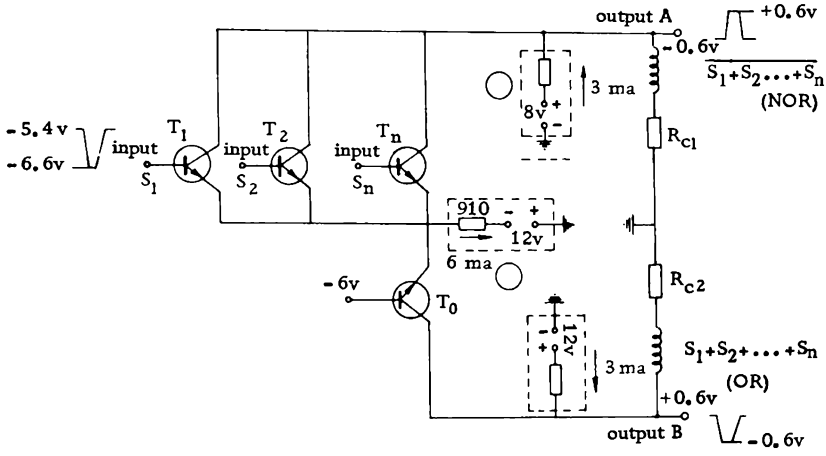


Fig. 116. OR and NOR circuits with n-p-n transistors.

applied to the base of T_3 and T_3 is cut off while T_4 is conducting. Thus, the flipflop is switched to the other stable state in which T_1 and T_4 are conducting while T_2 and T_3 are cut off.

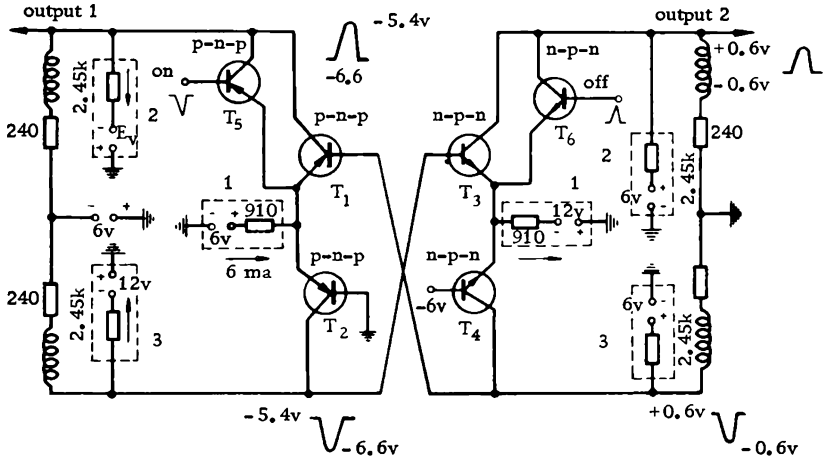


Fig. 117. Flipflop formed by two current switches.

The flipflop is reset to the initial state by application of a positive pulse to the base of T_6. T_6 is then set to the conducting state and T_4 is cut off; due to the cross coupling, the flipflop is reset to the initial state. The two outputs of the flipflop can be used for control of other current switches with *p-n-p* or *n-p-n* transistors.

8. VOLTAGE LIMITERS AND REGULATORS

Semiconductor circuits often require voltage limiting and regulation. This may be achieved by means of silicon diodes (of types D-808—D-813) with voltage rating at 7-14 v. These diodes are characterized by the so-called Zener effect in the reverse direction. If a voltage is applied to the diode in the reverse direction, the diode is cut off at low voltages. The reverse resistance of the diode is of the order of one megohm due to the "barrier layer" at the boundary of a p-n junction. The reverse current is determined only by thermal ionization, whereby a certain portion of the semiconductor atoms is ionized into negative charges (electrons) and positive charges (holes). The positive charges are displaced toward the negative pole of the voltage source and the negative charges toward the positive pole. The greater the voltage applied to the diode, the more rapidly the charge carriers are displaced within the semiconductor. At a certain voltage they may acquire such velocity that upon collision with neutral atoms of the semiconductor they split into electrons and positively charged ions. The voltage at which impact ionization begins is known as the threshold voltage. This ionization results in more charges which, in turn, ionize more atoms, etc.

As a result of absorption of charges within the semiconductor and their recombination, the ionization process does not continue indefinitely but is established at a definite level. At this level, a slight increase in the external voltage beyond the threshold voltage causes a considerable increase in the reverse current through the diode. Fig. 118 shows the voltage-current characteristic of a silicon diode, which shows that in the presence of a reverse voltage above the threshold (about -12 v) the reverse current rises abruptly, making a sharp bend. The curve almost parallels the ordinate axis. In addition, the reverse resistance of the silicon diode drops from a value of the order of one megohm to a value of several ohms. In a silicon diode, this effect is reversible and is applicable to voltage limiting and regulation. In a germanium diode, when the reverse voltage is close to the threshold voltage thermal breakdown of the diode occurs, resulting in heat liberation and additional thermal ionization within the semiconductor. The influence of thermal ionization is negligible in a silicon diode; the reversible process occurs without diode breakdown due to the higher thermal stability of silicon.

The simplest voltage regulator (Fig. 119a) consists of a resistor R connected in series with a silicon diode D (in reverse connection). As long as the input voltage is lower than the threshold voltage of the regulator, diode D is cut off (its resistance is high) and the output voltage corresponds to the input voltage. When the input voltage reaches the threshold level, the diode resistance becomes extremely small and the current through the diode suddenly increases. With a further increase in input voltage beyond the threshold value, the output voltage remains almost constant since the current through the diode rises but its voltage does not

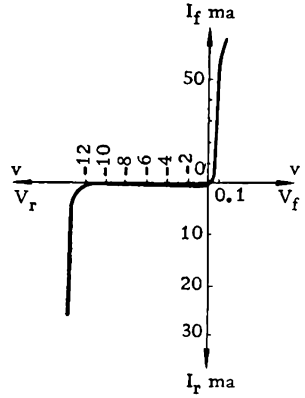


Fig. 118. Volt-ampere characteristic of a silicon diode.

change; almost the entire increment of input voltage drop is across resistor R. The regulation provided by the circuit increases as resistors R and R_L are increased in comparison with the dynamic impedance of the diode Z_d, which is

$$Z_d = \frac{dV_{in}}{dV_{out}} = 1 + \frac{R}{R_d}, \tag{24}$$

where R_d is the equivalent internal resistance of the diode. The regulation factor K of the circuit is equal to the change in voltage at the input divided by the change in voltage at the output, i.e.,

$$K = \frac{dV_{in}}{dV_{out}} \cdot \frac{V_{out}}{V_{in}} = \frac{V_{out}}{V_{in}}\left(1 + \frac{R}{R_d}\right). \tag{25}$$

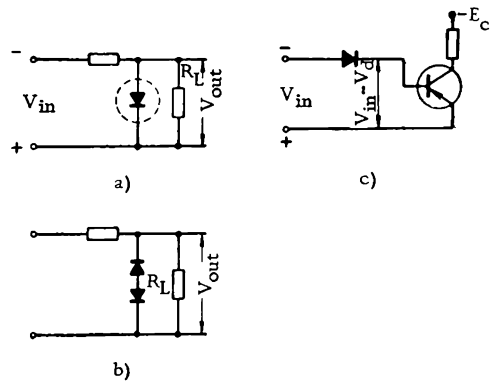


Fig. 119. Voltage regulator circuits.
a—nonsymmetrical limiting; b—symmetrical limiting; c—limiting the voltage at the base of a transistor.

Resistance R is defined as

$$R = \frac{V_{in} - V_{out}}{i_d + i_{out}}, \tag{26}$$

where i_{out} is the current through the diode (the Zener current). The output load resistance R_L is determined from the formula

$$R_L = \frac{R \cdot R_d}{R + R_d}. \tag{27}$$

For symmetrical voltage limiting, a circuit with two silicon diodes is used (Fig. 119b). Since in regulator operation the voltage across the silicon diode cannot be higher than the threshold voltage of regulation (determined from the diode voltage-current characteristic), that part of the external pulse at the input above this voltage is limited. In operation with short pulses it must be kept in mind that the intrinsic capacitance of a silicon diode at voltages close to threshold is of the order of 80-150 $\mu\mu$fd and through it a considerable capacitive current flows.

In order to limit the voltage at a transistor base (Fig. 119c), a silicon diode is connected between the input and the base, limiting base voltage to

$$V_{be} = V_{in} - V_t, \tag{28}$$

where V_t is the threshold voltage at the reverse-current break point of the diode voltage-current characteristic.

Such limiting is achieved, for example, in a flipflop circuit (Fig. 120) where, instead of an RC coupling network at the input, silicon diodes connected in the reverse direction are used. This permits circuit simplicity and limitation of the switching current at the transistor base. The latter permits us to reduce the deep saturation of the transistor; the saturation leads to stretching of the trailing edge of the pulse and decreases the switching frequency of the flipflop. Such flipflop circuits do not require special adjustment and are stable in operation. Switching of the flipflop is

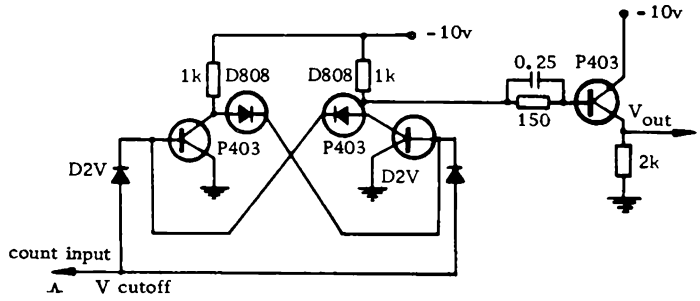


Fig. 120. Flipflop with diode limiters.

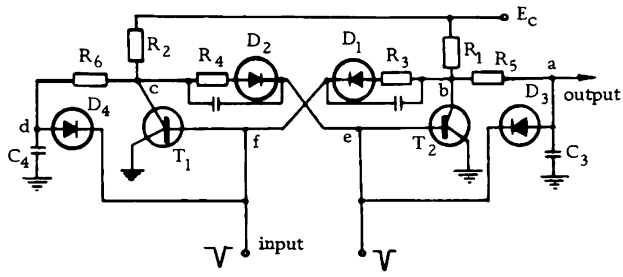


Fig. 121. Bistable multivibrator with diode limiters.

achieved by positive pulses with duration of 0.1-0.6 μsec applied to the base through ordinary D2V diodes. In order to improve the frequency and load characteristics of the flipflop, it is desirable to use an emitter follower at the output. By using type P403 transistors in this circuit, we may obtain a pulse leading edge of 0.1 μsec and trailing edge of 0.3 μsec. In the bistable multivibrator circuit (flipflop) in Fig. 121, silicon diodes D_1 and D_2 are used to limit the collector voltage applied to the bases of T_1 and T_2 over the feedback loop. They are connected in series with coupling resistors R_3 and R_4; the voltage at the base is equal to the difference between $E_c - V_{reg}$ where V_{reg} is the regulation level in volts. Diode D_1 has a lower regulation level than D_2. When T_1 is conducting current is being drawn from the base circuit of saturated T_1. Transistor T_2 is cut off since the potential at point c is close to zero and point b is at a high negative potential close to $-E_c$. Silicon diodes D_3 and D_4 serve to short the multivibrator circuit when the flipflop output voltage reaches a certain value. Capacitor

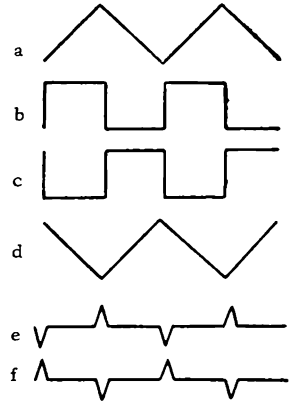


Fig. 122. Voltage waveforms (at points a, b, c, d, e, f of the circuit in Fig. 121).

C_3 is charged through resistors R_1 and R_5 until point a reaches the regulation level of diode D_3. Thereupon, the current passing through D_3 and the base of T_2 is close to zero, driving T_2 to saturation, and transistor T_1 is cut off since the potential at point b approaches zero. With T_1 cut off, point c is at a high negative potential (Fig. 122). Capacitor C_4 is charged through resistors R_2 and R_6 up to the regulation level of diode D_4. At the same time capacitor C_3 is discharged through resistor R_5 and transistor T_2. When the voltage at capacitor C_4 exceeds the regulation level of diode D_4 the multivibrator will again return to the initial state and capacitor C_4 will be discharged, C_3 will be recharged, etc.

Thus, the regulation level of silicon diodes D_3 and D_4 determines the moment at which the multivibrator is triggered.

9. DIODE AND DIODE-TRANSFORMER COINCIDENCE AND INHIBIT CIRCUITS

Digital computers employ two types of coincidence and inhibit circuits. In the first type, multiple coincidence and inhibit operations are achieved by a diode circuit with common resistor R (Fig. 123) and the output is applied to a transistor pulse amplifier through a stepdown transformer TR_1. Noise pulses are eliminated by a small negative bias E_B blocking diode D_5. The use of transformer TR_2 (with ferrite core) in the collector circuit of the transistor amplifier gives short pulses of both polarities at the output.

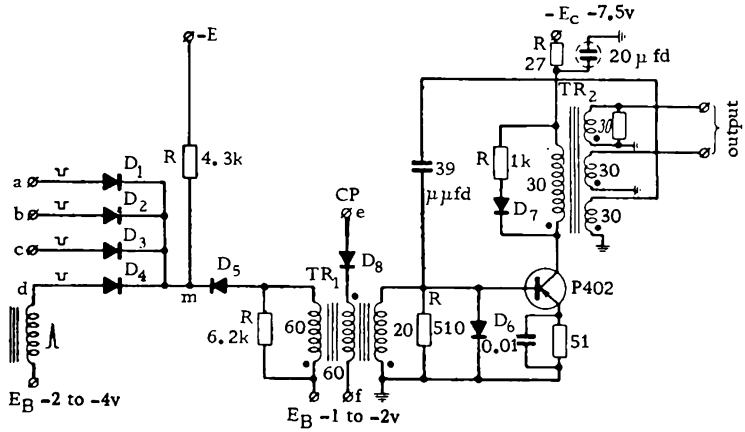


Fig. 123. Coincidence and inhibit circuit with pulse amplifier output.

In the absence of negative signals at the inputs of the coincidence circuit (diodes 1, 2, 3), diodes D_1—D_4 are conducting. Almost all the voltage drop is across resistor R and the potential at point m is close to zero. The inhibit circuit uses diode D_4, which is normally

cut off by negative bias $E_B = -4$ v. In the presence of a positive inhibit signal in the winding of the input transformer, diode D_4 conducts and inhibits the potential arriving at point m due to the coincidence circuit.

The AND coincidence circuit consists of diodes D_1-D_3. When negative signals are applied to the inputs of the AND circuit, the diodes are cut off; a voltage is induced across the transformer TR_1. Any diode may be used as the inhibit input, and one of the input diodes may be used for gating incoming pulses. The d-c voltage $-E$ may be replaced by a gating pulse. For normal operation of the logic circuit, the following conditions must be satisfied:

a) The internal resistance of the signal source must match the input resistance of the transistor;

b) In the absence of signals the load must be bypassed, i.e., the inequality

$$|r_i + r_d| \ll \left| \frac{R \cdot z_L}{R + z_L} \right|, \tag{29}$$

where r_i is the internal resistance of the signal source,
r_d is the diode forward resistance,
z_L is the load impedance at the output (of transformer TR_1);

c) Upon arrival of signals at the inputs of the AND circuit, the diodes must not bypass the load, i.e., the inequality

$$|r_{dr}| \gg \left| \frac{R \cdot z_L}{R + z_L} \right|, \tag{30}$$

where r_{dr} is the diode reverse resistance.

The output resistance of the AND circuit consists of the parallel resistance of R and of the reverse resistances of the diodes (i.e., it is actually equal to R, since the reverse resistance of the diodes is greater by several orders of magnitude).

The AND circuit operates in two conditions: (a) shunting of the load by the low resistance of the diodes and internal resistance of the pulse source; (b) with the diodes cut off, the voltage being divided between R and z_L.

The value of R must be as large as possible in order to decrease the voltage drop at $r_i + r_d$. On the other hand, matching of R with the input resistance of the transistor and operation in the second of the above conditions require a decrease in R.

Selection of R and of supply voltage E is based on the fact that diodes of type D9D in pulse operation with a current of the order of 1 ma have a resistance of 200-300 ohms. The input resistance of a P402 transistor with load of 130 ohms is approximately 50 ohms. The voltage at the base of the transistor in the conducting state must be approximately 0.6-0.8 v. The P402 transistor amplifier uses a small amount of positive feedback through a 39-$\mu\mu$fd capacitor, so that the differentiated pulse contributes to saturation of the transistor during passage of the leading edge of the pulse, while during passage of the trailing edge a positive pulse is applied

to the base, facilitating diffusion of the minority carriers within the base and preventing stretch of the pulse.

Resistor R (51 ohms) in the emitter circuit is a negative feedback resistor, stabilizing operation of the transistor and limiting the steady-state component of collector current in large-signal operation. The 0.01-μfd capacitor shunts the resistor and thereby decreases the resistance in the emitter circuit during pulse passage. Diode D_6 shunts voltage overshoots in the base-emitter circuit.

Diode D_7 with a series-connected resistor shunts voltage overshoots in the winding of collector transformer TR_2 and improves the pulse shape at its output. One of the windings of the base transformer with diode D_8 serves to form a diode-transformer circuit for coincidence of pulses at inputs e and f as described subsequently.

A disadvantage of this coincidence circuit is the large value of spurious signal at the output of the diode coincidence circuit (20-30%) and the low value of current in the primary winding of base transformer TR_1, limited by the relatively high value of resistance.

These drawbacks are largely eliminated in diode-transformer coincidence (Fig. 124) and inhibit circuits. In the schematic of Fig. 124, diode D_1 is normally cut off by bias voltage $-E_B$. If the voltage induced in the secondary winding of the transformer by pulses v_A or v_B is less than $|E_B|$, in the presence of only one pulse v_A or v_B diode D_1 does not conduct and the voltage at the output is equal to zero. If pulses v_A and v_B are applied to both inputs simultaneously, diode D_1 conducts since the voltage induced by pulse v_B in the secondary winding of the transformer overcomes the cutoff voltage $-E_B$. A change in the number of turns in both primary and secondary windings does not affect the output pulse amplitude, as the latter is determined by the turns ratio and input pulse amplitude. A change in the number of turns merely widens or narrows the output signal. A substantial advantage of this coincidence circuit is the possibility of obtaining considerable power at its output. With a load of 1 kilohm or more at the output of TR_1, the pulse amplitude is independent of change in load. With a 500-ohm load, the output pulse amplitude decreases by 30%. If the output impedance of the transformer is matched with the load impedance, the pulse amplitude may remain constant at low load resistances. Another important advantage is the almost complete absence of spurious signals in the presence of properly chosen input signal amplitudes. The amplitude of voltages v_A and v_B in the transformer windings must not be greater than the voltage E_B which cuts off the diode. With $v_A = v_B > E_B$ there appears a spurious signal equal to the difference $E_B - v_A$ or $E_B - v_B$. With $v_A = v_B < E_B$ the spurious signal is equal to zero but the input signal amplitude is correspondingly decreased by the value of the difference $E_B - v_A$ or $E_B - v_B$.

Thus, one disadvantage of the circuit is the dependence of the output signal and spurious signal input pulse amplitudes with fixed value of E_B. however, this drawback may be largely eliminated if

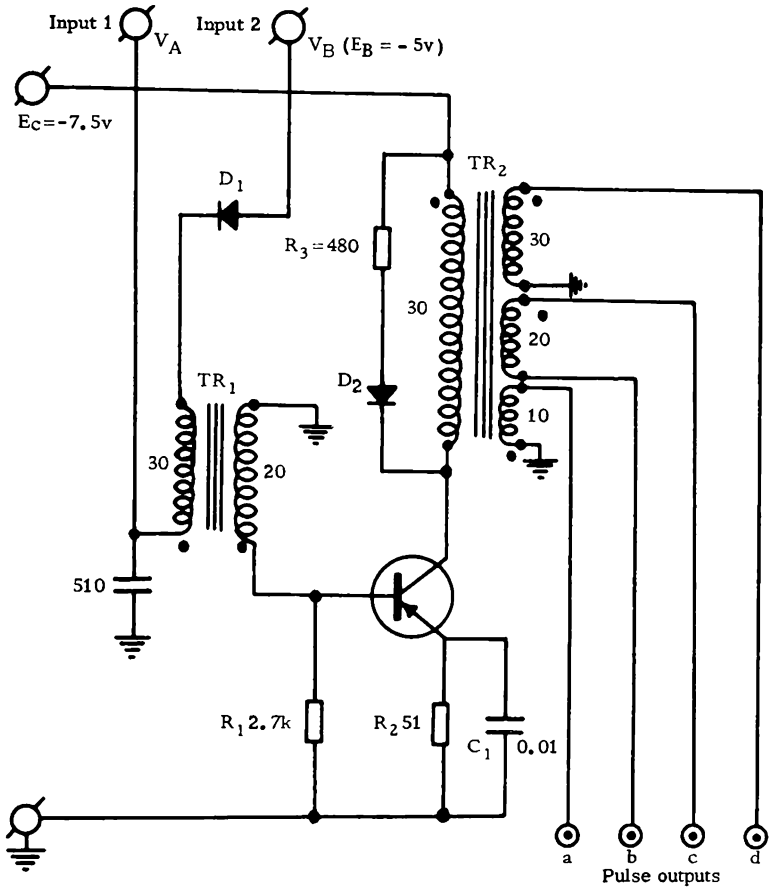


Fig. 124. Diode-transformer coincidence circuit.

the d-c source E_B also serves as the source of voltage applied to the circuit in order to obtain pulses V_A or V_B at the input of the coincidence circuit. In this case, the amplitude of pulses V_A and V_B will be determined by the value of E_B and a change in supply voltage will have little effect on operation of the coincidence circuit.

The diode-transformer inhibit circuit in Fig. 125 is constructed with a three-winding transformer TR_1 where windings 1 and 2 are connected in opposite polarities so that upon passage of current the resulting flux is equal to zero. A negative signal A is stored for a certain time by means of capacitor C_1. This same signal A, but of positive polarity, is applied to the second input of the diode-transformer coincidence circuit with diode D. If inhibit signal B is absent, a pulse appears at the output of the transistor amplifier. If an inhibit signal is present, the current in winding 2 of the transformer is opposed to the current in winding 1 and no signal appears at the output. Inhibit signal B must arrive before signal A. It is

stored by capacitor C_2, which facilitates phasing of both signals in time. In the absence of signal A, inhibit signal B does not arrive at the output, since diode D is cut off by the bias voltage $-E_B$.

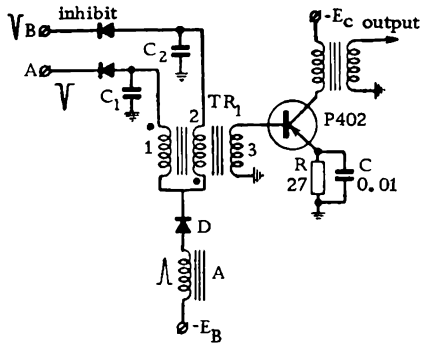


Fig. 125. Diode-transformer inhibit circuit.

We may also use pulse circuits with transistor amplifiers as coincidence and inhibit circuits not requiring time coincidence of input pulses. The pulse at one of the outputs is temporarily "stored" until the arrival of a pulse at the second input. This "storage" is achieved in the coincidence circuit of Fig. 126 by the accumulation of minority carriers upon the application of an input pulse to the transistor base; these carriers are "removed" from the base region upon passage of a current pulse in the collector circuit when a pulse is applied to the second input. The permissible pulse "storage" time at the transistor base is usually chosen of the order of 0.1-0.2 μsec.

In the inhibit circuit, "storage" of the inhibit pulse is achieved at the base of transistor T_1 (Fig. 127). For reliable operation of the inhibit circuit the inhibit pulse must possess larger width than the pulse at the second input; this is usually achieved by the use of inhibit-pulse "stretchers." Due to storage of the inhibit pulse at the base of T_1, the pulse applied to input B passes through the collector circuit of the transistor also in the case when the inhibit pulse is applied to input A before the pulse at input B. In the absence of an inhibit signal, the transistor is cut off and, through the primary winding of the transformer and the resistor in the emitter circuit, there flows a current generated by the voltage pulse at input B. In the presence of an inhibit signal at input A the transistor is driven to saturation and diode D_1 is cut off by the voltage drop of -4 v at the emitter resistor. Hence the signal at output B does not pass through the transformer winding.

If the inhibit signal at input A is given in the form of a negative potential which is applied, for example, from a flipflop output, the inhibit circuit of Fig. 128 is used, where the potential at input A becomes clock pulses CP by means of the emitter follower of T_1. In the presence of a negative voltage at input A, transistor

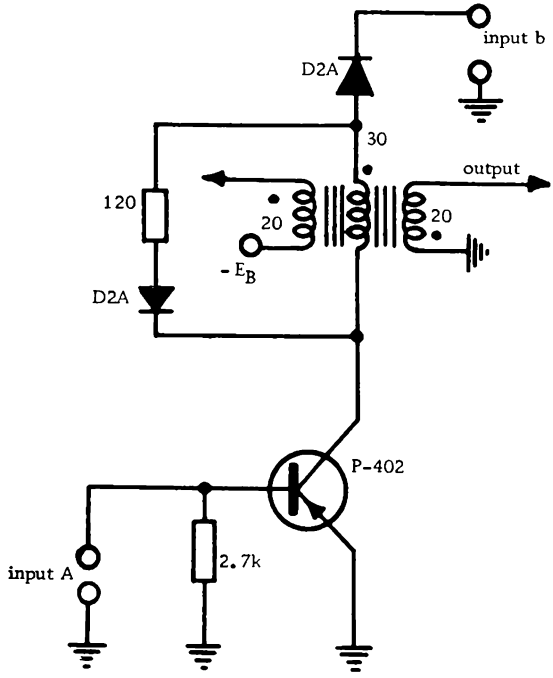


Fig. 126. Gate amplifier (coincidence circuit).

T_1 is conducting and the CP pulses pass through the coincidence circuit consisting of a diode and transformer TR_1. If the potential at the input A is zero, T_1 is cut off, the voltage drop at the resistor in the emitter circuit is equal to zero, and diode D_1 is cut off by

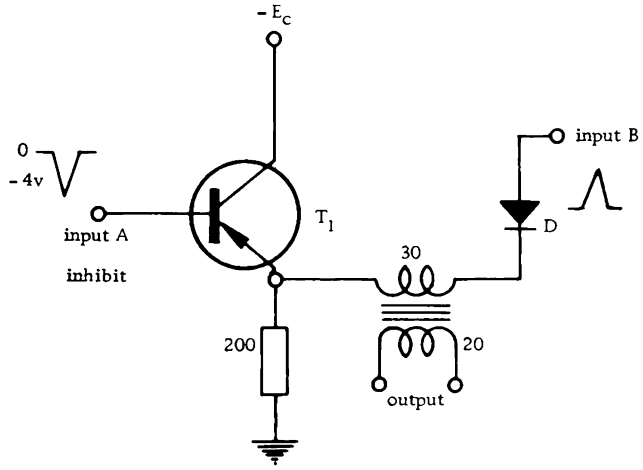


Fig. 127. Inhibit circuit with emitter follower.

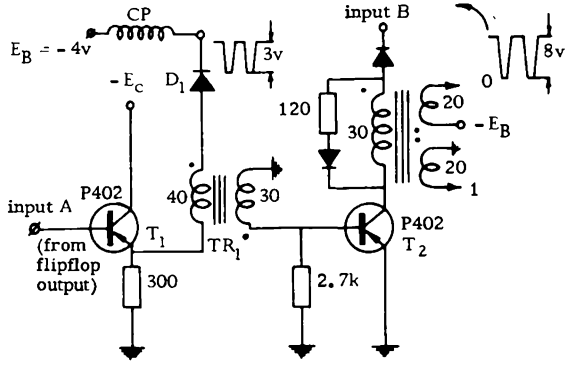


Fig. 128. Inhibit circuit.

bias voltage $E_B = -4$ v (applied to the winding of the CP pulse source). Hence the CP pulses do not pass through the winding of transformer TR_1. The pulses from the output of transformer TR_1 are applied to the base of T_2 and "stored" therein until arrival of a pulse at input B (the storage time is of the order of 0.1-0.2 μsec).

Transmission of only the negative voltage pulses at input B to the collector circuit is achieved by diode D_2. Diode D_3 and resister R (120 ohms) suppress the "overshoots" of self-induced emf occurring upon turn-off of the transistor.

Another version of the inhibit circuit is that given in Fig. 129, wherein the inhibit pulse is applied to the emitter circuit (input A).

In the absence of an inhibit pulse the transistor is cut off, and the pulse at input B passes through capacitor C and the transformer winding. A voltage pulse is then obtained at the output of transformer TR. If an inhibit pulse is applied to input A, the transistor conducts and a current passing through diode D_1 and the transformer

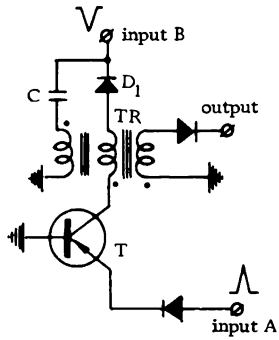


Fig. 129. Circuit with inhibit pulse applied to emitter.

secondary produces a flux compensating the flux in the primary winding. Hence the transformer output voltage is equal to zero.

10. DYNAMIC FLIPFLOPS

There are several types of dynamic flipflops in which storage can be achieved by capacitors, delay lines, or a property of the transistor itself—its ability to "store" an input pulse for a certain time. The dynamic flipflop is a memory cell where, for code 1, there is a continuous sequence of pulses of a definite frequency at its output. For code 0, there is no pulse at the output. Thus, the dynamic flipflop is itself a type of pulse generator upon receiving a code 1 input.

The block diagram of a dynamic flipflop with capacitor storage is shown in Fig. 130. The flipflop has two inputs: the first input is used for receiving the code 1 pulse and the second for the code 0 pulse. When a code 1 pulse is applied to input 1, it is first amplified by amplifier A and then stored in element SE (in this case a capacitor). As the capacitor is now charged, the capacitor voltage controls the AND gate, which allows passage of CP pulses arriving from an external pulse generator with a definite frequency. The pulse output of the AND gate is fed back to the amplifier, and thus a loop is formed for circulating the pulse. It suffices to apply at the input 1 a single pulse representing code 1 in order to obtain a continuous sequence of amplified CP pulses. This will occur until a pulse representing code 0 is applied to the storage capacitor and completely discharges the charge on the capacitor. Thereafter, the AND circuit will no longer pass the CP pulses to circulate in the loop.

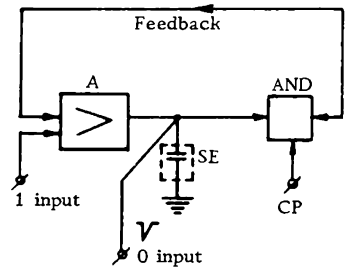


Fig. 130. Block diagram of dynamic flipflop with a storage capacitor.

Figure 131 shows the schematic of a dynamic flipflop (with a P502A transistor) with storage capacitor. The flipflop is set to the 1 state by the application of a negative pulse (-6 v) to charge storage capacitor $C_1 = (510\,\mu\mu\text{fd})$. The capacitor voltage is fed back to one of the inputs of a diode-transformer coincidence circuit consisting of the primary winding of transformer TR_1 and

diode D_1. Positive pulses from a special pulse generator with frequency of 1 Mc are applied to the second input. In the absence of CP pulses diode D_1 is cut off by the bias voltage $E_B = -6$ v applied to the transformer winding which serves as the source of the CP pulses. If capacitor C_1 is charged, the next CP pulse discharges the capacitor through transformer TR_1 and drives T_1 (a P502 n-p-n transistor) to saturation. Through diode D_2, the transistor-amplified negative pulse again charges capacitor C_1, which is again discharged at the next CP pulse; thus the flipflop operates according to the clock-pulse cycle. The time constant for discharge of the capacitor through cut-off diode D_2 is considerable; the capacitor is not discharged within 0.5-1 μsec. In order to interrupt the delivery of pulses at the flipflop output, it is necessary to discharge capacitor C_1 by a positive pulse through diode D_3.

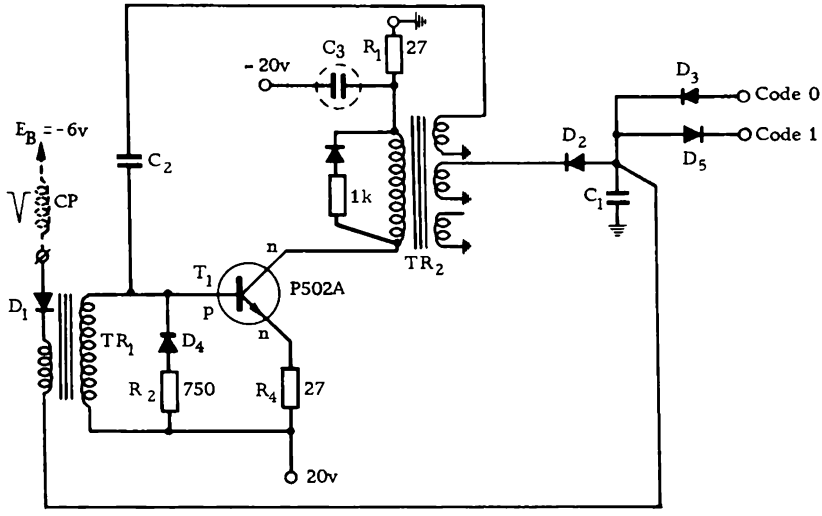


Fig. 131. Schematic of dynamic flipflop with storage capacitor.

As a criterion in evaluating the stability of flipflop operation we may take the range of variation in the CP bias voltage for different operating conditions of the flipflop. Actually, this corresponds to a change in amplitude of the CP pulse. In testing the flipflop, the spontaneous triggering bias (minimum bias) and the clearing bias (maximum bias) were established. The operating region of the flipflop must lie within these boundaries; the wider this range, the more stable the flipflop operation.

In order to increase operating stability of the flipflop it is necessary to drive the transistor to deeper saturation, but this causes a stretch of the output pulse. Positive feedback through capacitor C_2 can be used to increase flipflop stability, yet avoiding to a certain degree the widening of the pulse.

The use of positive feedback would lead to speeding up of flipflop operation. But this would cause an undesirable lengthening of

the trailing edge, since the base would be oversaturated with minority carriers toward the end of the pulse. Thus, transformer coupling causes stretching of the pulse to 0.8-1 μsec.

Positive differentiating feedback through capacitor C_2 is used in the flipflop. At the leading edge of the pulse this feedback contributes to transition of the trigger to saturation. The feedback is absent at the top of the pulse. At the trailing edge of the pulse a positive voltage is applied to the base to prevent stretching.

Introduction of positive differentiating feedback into the flipflop leads:

a) To wider stability limits in flipflop operation. For example, for a flipflop using a type P402 transistor without feedback the minimum and maximum values of bias were - 4 v and - 5.5 v, respectively (i.e., a permissible variation of ±16% in bias voltage). In using positive feedback through capacitor C_2 the minimum and maximum values of bias were - 3 v and - 7 v, respectively (i.e., a permissible variation of 40%).

b) To improvement in the trailing and leading edges of the output pulses.

c) To a certain decrease in flipflop sensitivity, an advantage in that the flipflop then possesses greater noise immunity. The decrease in sensitivity is due to the differentiating circuit being an additional load on the transistor;

d) To improved recombination of minority carriers in the base upon termination of the CP pulse.

e) To less critical dependence of the flipflop on the β current gain of the transistor. Transistors with β from 11 to 90 were found to operate in the circuits.

f) To a higher frequency limit of flipflop operation. The upper frequency limit is determined by the time constant of capacitor discharge through the diode. The positive feedback increases the operating speed of the circuit.

The feedback pulses are taken from the winding of TR_2 and are differentiated by the circuit consisting of C_2 and the input resistance of transistor T_1. R_2 and D_4 serve to bypass negative overshoots. Due to its non-linear characteristic, diode D_4 protects the base against positive voltage breakdown by maintaining a positive base potential of the order of +0.3 v, which is sufficient to cut off the transistor and to improve the pulse shape. In addition, together with resistor R_2, diode D_4 bypasses the voltage overshoot in the base transformer. Electrolytic capacitor C_3 and resistor R_1 are used as a filter in eliminating external noise.

For the circuit of the dynamic flipflop in Fig. 131 the following experimental data have been obtained:

a) The normal operating limits of the flipflop are maintained with a change in bias voltage in the CP winding from -4.5 to -7.5 v.

b) The flipflop circuit is not critical with respect to spread in β gain of transistors, which varied from 10 to 30 and higher.

Replacement of the transistors in the circuit did not disturb normal operation. Operation of the flipflop was also normal upon application of single pulses. It was triggered upon application

of a single 0.5-μsec pulse to its input and cleared by a similar pulse of opposite polarity.

c) Reliable operation of the flipflop was achieved at temperatures ranging from -50 to +50°C and higher.

d) The circuit is not critical with respect to external noise (voltage surges upon a change in load, etc.).

e) The circuit is not critical with respect to a change in parameters of its components. The transistors operate in the saturation region. The use of transformer coupling eliminates the effect of slow changes in transistor characteristics.

f) Pulses of code 1 and 0 may arrive at the input of the flipflop at any time and do not coincide in phase with the *CP* pulses.

g) The flipflop circuit consists of a small number of components: one transistor, six diodes and two miniature ferrite-core transformers.

h) Operating frequency of the flipflop and of the clock generator may be varied over a wide range from 1 to 1.5 Mc and higher.

It is desirable in some cases to have a dynamic flipflop with voltage level output. The presence of voltage level output permits control of the gates regardless of the timing of pulses. The emitter follower in Fig. 132 with type P402 transistor is a dynamic flipflop with voltage level output. Capacitor *C* is charged by a negative pulse for code 1 and discharged by a positive pulse for code 0. The voltage of capacitor *C* is applied to the transistor base.

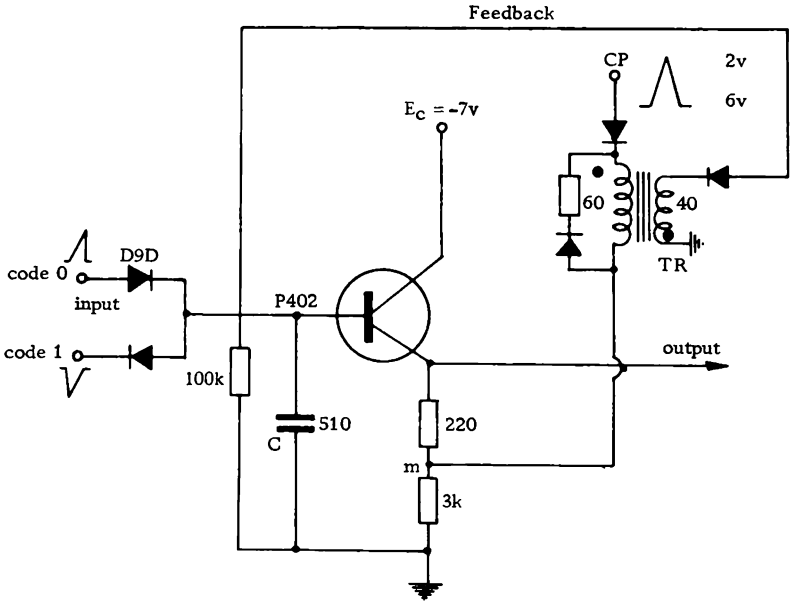


Fig. 132. Single-transistor dynamic flipflop with pulse output and voltage level output.

When capacitor C has a potential close to zero (or positive) transistor P402 is cut off and the voltage at the emitter follower output is close to zero. With the arrival of a negative pulse at the emitter follower input (with pulse amplitude of 4 v), capacitor C is charged and transistor P402 is in the conducting state. In order to keep capacitor C charged, pulses from the diode-transformer coincidence circuit are fed back. The CP pulses pass through the primary winding of transformer TR only when transistor P402 is conducting. In this case the bias voltage applied to the winding of the CP generator ($E_B = 6$ v) is compensated by the voltage at point m of the divider. With the component values given in the figure, the flipflop operates at a pulse-repetition frequency of 1 Mc.

If another transistor T_2 is connected in the feedback loop as an amplifier, we obtain the circuit of a dynamic flipflop (Fig. 133), having both a voltage level output 1 from the emitter follower and a pulse output 2 from the transistor amplifier. Moreover, power-amplified pulses are available at output 3. The voltage divider consisting of resistors R_2 and R_5 is used to increase the voltage charging capacitor C_1 through diode D_2. The operating stability of a flipflop with two transistors is considerably higher than with one. The ability of a transistor to "store" for a certain time a pulse arriving at its input (just as in a diode amplifier the input pulse "ignites" the diode) permits the use of this property in a dynamic flipflop as shown in Fig. 134 without a storage element in the form of a capacitor or delay line. In this schematic, the pulse representing code 1 is applied to the emitter-base circuit of the transistor and gives rise to the accumulation and storage of minority carriers (holes) in the base, requiring a certain period of time for their recombination. If at any time an "igniting" pulse is applied to the emitter, the negative CP pulse at its collector, being further displaced in time, produces at the winding of transformer TR a collector current pulse. Through the feedback loop the emitter is again "ignited" with each occurrence of the CP pulse. The result is that the transistor retains its conducting state until a negative pulse representing code 0 is applied to the collector and completely "extinguishes" the transistor. The shape and duration of pulses at the output and in the feedback loop are completely determined by the shape and duration of the CP pulses; hence, "stretching" of the pulse trailing edge, which usually occurs in flipflop circuits, has no effect here. Hence a dynamic flipflop with type P6G transistor in this circuit permits achieving a pulse repetition frequency of 2-3 Mc. The pulse train may be decreased to a single pulse at the flipflop output with subsequent extinction of this pulse. Advantages of this circuit are its simplicity and the absence of storage components therein. A disadvantage is the need for relatively high-power clock pulses for the collector supply of the transistor. Permissible variations in the CP voltage source in this circuit are of the order of 20%. The circuit is not critical with respect to variation in parameters of the transistor and other components. Dynamic flipflops may also be made with diode elements. However, a significant disadvantage of the diode flipflop is the necessity of having a large

"igniting" pulse current and using high-power CP sources in order to obtain a high-power pulse at the flipflop output.

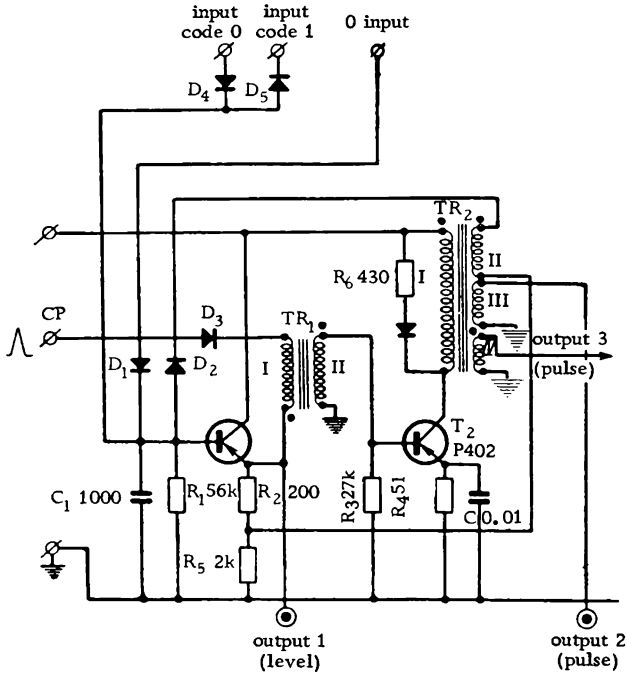


Fig. 133. Two-transistor flipflop with pulse output and voltage level output.

A dynamic flipflop may also employ a delay line as the storage element (Fig. 135). It differs from a dynamic flipflop with capacitor in that the former circuit has a gate to inhibit transmission of pulses from the feedback loop to the amplifier input upon application of

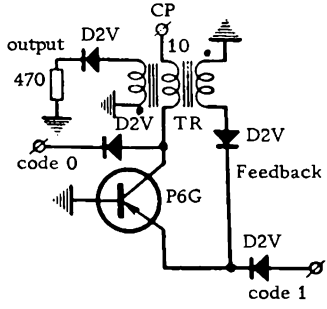


Fig. 134. Dynamic flipflop with storage in the transistor itself.

the inhibit pulse (representing code 0). The code 1 pulse is applied to the input of amplifier A. The amplified pulse passes through delay line DL in a period of time equal to the interval between two CP pulses. Upon coincidence of the pulse from the delay line with the next CP pulse, the latter passes through the coincidence gate, which also functions as a shaper. If there is no inhibit pulse, the pulse from the output of the AND gate is fed back through the feedback loop to the amplifier input. Pulses reappear at the amplifier input until an inhibit pulse representing code 0 is applied. The inhibit pulse interrupts transmission of the clock pulses to the coincidence circuit and halts circulation of the pulses at the flipflop output.

An important disadvantage of the dynamic flipflop with delay line is its sensitivity to a shift in phase of the pulse delayed in the line as well as of the CP pulse and the necessity for close coincidence of the input and CP pulses.

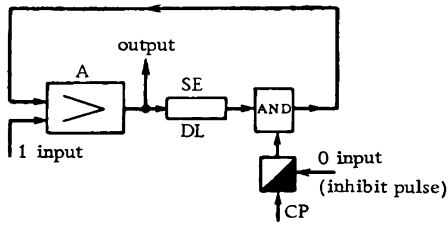


Fig. 135. Block diagram of dynamic flipflop with delay line.

11. HIGH-POWER DYNAMIC FLIPFLOPS

In the various units of a digital computer, particularly in the control units, flipflops must control many switching circuits. For this purpose high-power amplifiers are provided in the flipflop circuits. Figure 136 shows a more economical high-power dynamic flipflop in which two series-connected transistor amplifiers control a large number of gating and switching circuits.

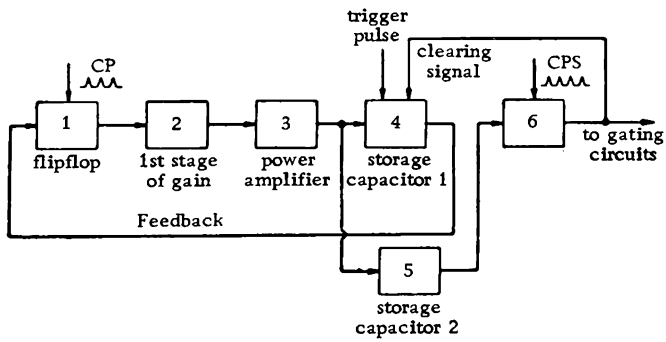


Fig. 136. Block diagram of high-power dynamic flipflop.

The high-power dynamic flipflop shown in Fig. 137 is in principle similar to that of a low-power single-transistor flipflop. Here, there is a second stage with a power transistor of type P601B and a preamplifier stage with a P402 transistor. CP pulses are applied to the input of a diode-transformer coincidence circuit through the primary winding of transformer TR_1 after storage capacitor $C_1 = 1000$ $\mu\mu$fd is first charged by a trigger pulse through diode D_8. To one of the output windings of transformer TR_3 (connected in the collector circuit of P601B), a second capacitor C_2 (510 $\mu\mu$fd) is connected through diode D_{10}; this capacitor stores the pulse arriving at the input of the diode-transformer coincidence circuit.

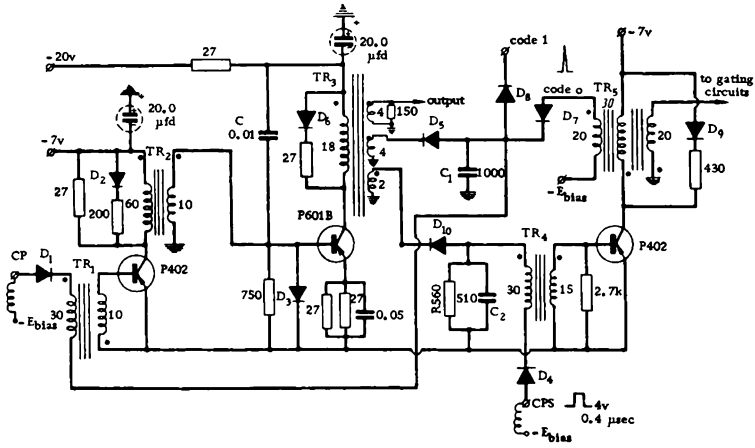


Fig. 137. High-power dynamic flipflop.

This coincidence circuit is required for use of the flipflop in counting circuits. Capacitor C_2 discharges through the winding of transformer TR_4 and diode D_4 upon application of a CPS pulse (shifted relative to the CP pulse by 0.5 μsec). A positive pulse through diode D_7 discharges the capacitor and clears the flipflop. Simultaneously a negative pulse from the other winding of collector transformer TR_5 actuates the next flipflop of the counter.

Resistor R (560 ohms) shunts diode D_{10} and dissipates the charge on capacitor C_2. The value of this resistor differs according to the counting circuit in which the flipflop is used. In a counting circuit operating at frequency of 1 Mc this resistor has a value of 560 ohms; in a counting circuit where the flipflops have a switching frequency of 50 kc resistor R is 2.7 kilohms; at even lower switching frequency, R is 20 kilohms.

In order to prevent false triggering of the flipflop due to noise impulses between the supply sources and their associated circuits, filters with 20-μfd capacitors (tubular electrolytic) are inserted.

Capacitor C (0.01 μfd), bypassing the base-collector circuit of transistor P601B, serves as a pulse shaper.

Each of the transformers TR_2, TR_3 and TR_5 in the collector circuits of the transistors is shunted by a diode and a series resistor in order to suppress pulse overshoots in the transformer and thus improve the pulse shape.

The above described high-power dynamic flipflop is extremely stable both in single-pulse operation and in pulse "train" operation with large variations of supply voltages and over a wide range of ambient temperature (Table 8).

TABLE 8

Operation of Dynamic Flipflop

Designation	Parameter values			
	Normal	Min.	Max.	Permiss. Var., %
Collector voltage, E_{c1}	-7 v	-6 v	-8 v	±14
Collector voltage, E_{c2}	-20 v	-17 v	-23 v	±15
Bias voltage CP, E_{CP}	-2.7 v	-1.8 v	-3.6 v	±33
Bias voltage, E_{CPS}	-1.4 v	-0.9 v	-1.9 v	±36
Capacitor clearing bias voltage, E_B	-4.3 v	-2.4 v	-6.2 v	±47
Ambient temperature	+20°C	-50°C	+60°C	—
Permissible variation in gain for:				
a) P402 transistors	From 20 to 150			
b) P601B transistors	From 20 to 100			

12. HIGH-POWER PULSE GENERATORS AND SHAPERS

Digital computer control units must possess high-power sources of timing pulses for bit time, word time, etc. which must drive a large number of logic circuits. For example, the pulse supply of dynamic flipflops requires a high-power clock pulse generator (CP). The clock pulse generator of the computer consists of two parts: a low-power clock and a power amplifier. Fig. 138 shows the schematic of such a clock-pulse driver with frequency of 1 Mc. It consists of two transistor stages (T_1 and T_2). The first stage (a P16A transistor), consisting of a Colpitts circuit with capacitors C_3, C_4 and C_2 and the primary winding of transformer TR_1, is the oscillator proper, providing sine waves with a frequency of 1 Mc. The generation of sine wave is stabilized by a negative bias applied to the base of transistor T_1 through a divider consisting of R_1 and R_2. Resistor R_3 serves as a self-bias resistor and is bypassed by capacitor C_4. Capacitor C_5 is a coupling capacitor

and prevents the passage of the direct component of collector current through the primary winding of transformer TR_1. The collector load is r-f choke Ch_1 with ferrite core (dimensions $12 \times 5 \times 5$ mm with a 120-turn winding). The self-excitation frequency is the natural frequency of the oscillator circuit consisting of the primary winding of transformer TR_1 and capacitors C_3 and C_4. With the given circuit parameters this frequency is 1 Mc.

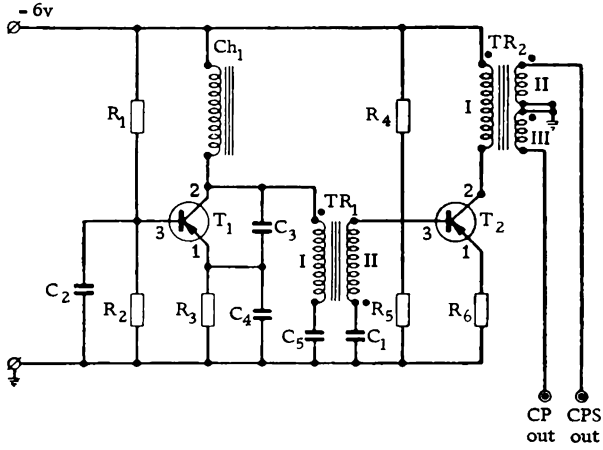


Fig. 138. Clock-pulse driver with frequency of 1 mc.

The secondary winding of transformer TR_1 feeds the base-emitter circuit of T_2, which is a current amplifier. The collector circuit of T_2 includes two windings in which the two sine wave voltages have 180° difference in phase. One half-wave of the sinusoidal voltage is used for generation of the CP pulses and the other half-wave (with a time displacement of 0.5 μsec) for generation of the CPS pulses.

The power amplifier of the CP and CPS generators in Fig. 139 is a three-stage amplifier. The first two stages, using P402 transistors, serve for pulse shaping and current amplification. The negative half-wave of the sinusoidal voltage of the clock is amplified by transistor T_1 and the positive half-wave is not conducted. Overshoots occurring in the windings of transformers TR_1 and TR_2 during pulse shaping are bypassed by diodes D_1 and D_2 with series-connected resistors. At the output transformer TR_2 there is obtained a shaped pulse of negative polarity with duration of 0.3-0.4 μsec (leading edge 0.07 μsec, trailing edge 0.1 μsec) and at a frequency of 1 Mc. This pulse is amplified by a P601B power transistor T_3. The collector circuit of T_3 includes a ferrite-core transformer with higher power rating. The output windings of this transformer provide high-power voltage pulses of 2, 4 and 6 v with pulse currents of the order of several amperes. A constant bias voltage of -6 v is applied to one of the windings in order that

the output may be used to feed the diode-transformer coincidence circuits. Capacitor C_3 (1300 $\mu\mu$fd) improves the pulse shape. Capacitor C_4 (20 μfd) and resistors R_7 and R_8 form a filter for suppression of noise in the supply circuit.

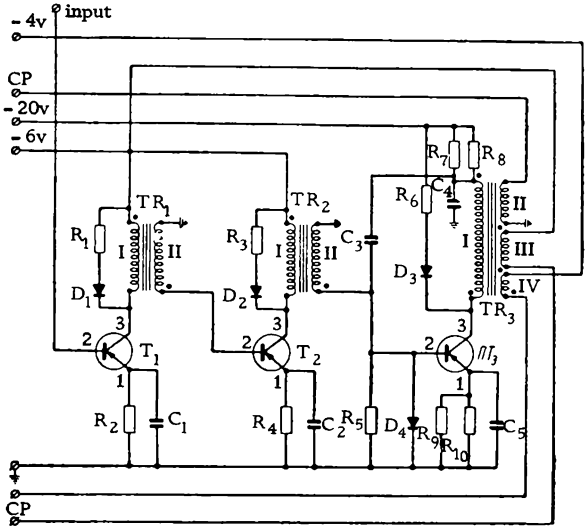


Fig. 139. CP and CPS power amplifier.

The power amplifier for the CPS generator is of the same type as above and differs only in that the voltage from the other winding of the sine wave generator (with 180° phase displacement) is applied to its input. The CP and CPS generators maintain stable operation in the presence of supply voltage variations of ±20 percent and within the temperature range from -50 to +50°C.

13. DYNAMIC FLIPFLOP SHIFT REGISTERS

Each stage of a high-speed flipflop shift register (Fig. 140) contains a dynamic flipflop with storage capacitor, having two inputs (1 and 0) and one output. Shift of the stored code from one stage to the next (from left to right in the circuit) is achieved by means of gates B_1 and B_2. Gate B_1 transmits the pulses from the shift bus to input 1 of flipflop FF_2 if the previous flipflop FF_1 is in the 1 state. If is is in the 0 state, anticoincidence gate B_2 transmits a clearing pulse to the 0 input of FF_2, since the inhibiting pulse for gate B_2 from the previous flipflop is absent in the presence of code 0. Gate B_3 (coincidence gate) transmits a clearing pulse to the input of flipflop FF_2 only when it is in the 1 state and it is actually necessary to clear it upon transfer of code 0 from the previous flipflop (having code 0). If FF_2 is already in the 0 state,

the clearing pulse is not transmitted by gate B_3 so that the flipflop capacitor is not charged by negative voltage. Transfer of the code from flipflop FF_1 to the input of FF_2 is achieved after application of the shift pulse with a certain small delay (e.g., 0.3 μsec for frequency of 1 Mc). This is necessary in order to transfer the code stored in FF_1 to the next stage FF_2, since the state of FF_1 may also change upon transfer of the code from the previous stage to its input. Intermediate storage of the code from the previous stage FF_1 to FF_2 is achieved by means of gate amplifiers B_4 and B_5, which transmit the stored pulses to the input of flipflop FF_2 at the moment of application of a control pulse from the bus with 0.3 μsec delay relative to the shift pulse.

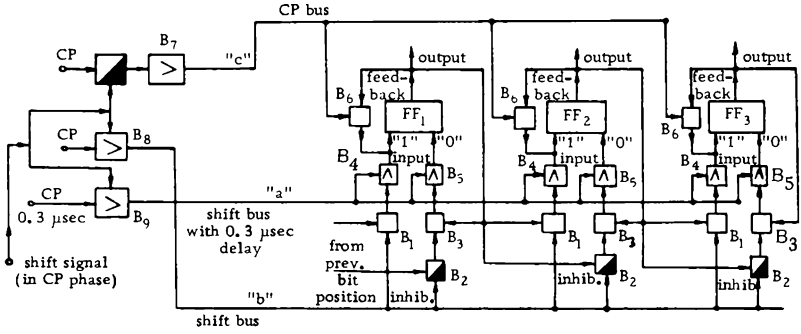


Fig. 140. High-speed flipflop shift register.

Storage is achieved due to the fact that the code pulses from gates B_1 and B_2 are applied to the bases of transistor amplifiers B_4 and B_5 and are stored in the form of the conducting state of the transistors until arrival of the delayed pulses at their collectors.

Each flipflop has a feedback loop for prolonged storage of code 1 with gate B_6 transmitting the CP pulses. Upon application of the shift signal, the passage of CP pulses to bus "c" is inhibited by gate B_7 and the feedback loops of the register flipflops are interrupted. Gate amplifier B_8 then transmits the shift pulses to bus "b" and gate B_9 transmits the CP pulses to bus "a" with a 0.3-μsec delay. This transistor register permits shifting the number code in one cycle and does not require intermediate storage of the codes in delay lines.

Shift registers are often used for storage of parallel code and for conversion of the parallel code to serial code by shifting. For this purpose we may use the simpler dynamic flipflop register represented in Fig. 141. Here the transfer of the code to the dynamic

flipflop register ($FF_1 - FF_{18}$) is achieved in parallel to corresponding stages through gates $B_1 - B_{18}$ upon the transfer signal. The code of each bit position is stored by a dynamic flipflop and may be stored therein as long as required. Readout of a number stored in the register in the form of a serial code is achieved through gates $B_1 - B_{18}$, which are actuated in sequence by pulses $P_1 - P_{18}$. This same circuit may be used for the serial insertion and storage of a number code in the register and subsequent transfer in parallel to the computer memory. Such registers are widely used in the memory unit of the digital integrating computer and as lines for storage of increments from integrators and external input channels.

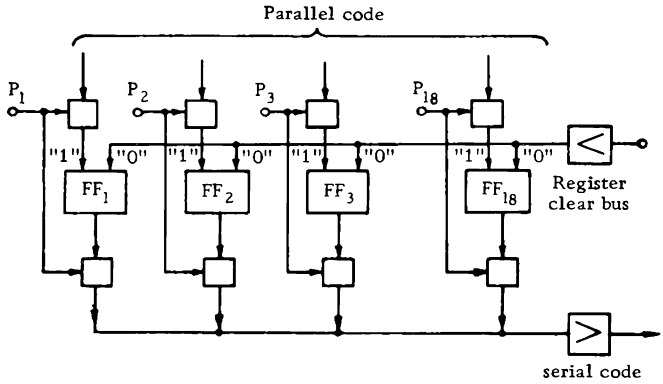


Fig. 141. Dynamic flipflop register.

14. COUNTING CIRCUITS

The control unit of the digital integrating computer must contain a counter to count the pulses for bit times and word times. The use of ordinary binary counters requires a decoder, but the output lines of the decoder do not have enough power to drive other logic circuits. Hence it is necessary to have a separate power amplifier at each output line. This considerably complicates the control unit of the computer. The use of a ring counter in combination with diode-transformer matrices permits a control unit which needs only a small number of power amplifiers.

Figure 142 shows a ring counter using dynamic flipflops. Each stage of the ring counter consists of a dynamic flipflop and a gated amplifier GA (a two-input AND circuit). Dynamic flipflop FF_1 is first set to the 1 state by means of start pulse SP. Gate GA_1 then permits passage of the pulse applied to the shift pulse line on to the next stage. In the meantime, a pulse at the "shift" line clears flipflop FF_1 and sets the next flipflop FF_2 to the 1 state. The input pulses to the "shift" line (CPS pulses) may arrive at any frequency up to the CP pulse repetition frequency. The CPS pulses

applied to the shift line are displaced in phase relative to the CP pulses by a half cycle (0.5 μsec with a CP frequency of 1 Mc). The ring counter is closed by feedback so that after the last flip-flop has been triggered the count begins again with the first flipflop.

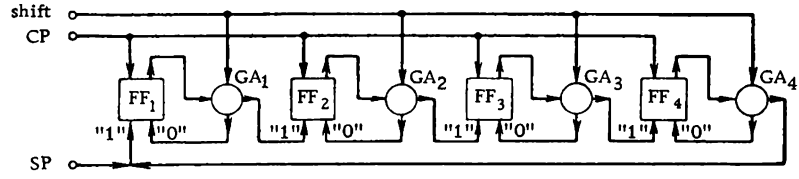


Fig. 142. Dynamic flipflop counter.

The pulse time diagram for a 4-bit ring counter is shown in Fig. 143.

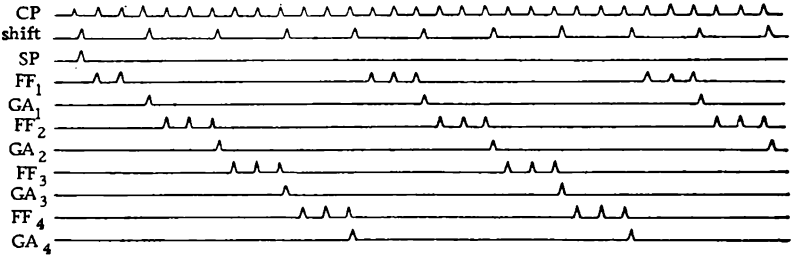


Fig. 143. Pulse time diagram for ring counter.

The above discussed circuit is designed to count the pulses in one direction. In certain cases it is necessary to count in both directions in order to add and subtract pulses.

For example, in a digital integrating computer, a reversible counter with transistor flipflops is also used as a counter for the increment pulses $\sum \Delta y$ of the integrand. A reversible counter is shown in Fig. 144; it has two inputs: add input (shift +) and subtract input (shift —), as well as a flipflop for the sign digit.

Pulses of code 1 arriving at the add input are counted by means of dynamic flipflops and gates B_1, B_2 etc, as was described above. Counting in the reverse direction (subtraction of pulses upon their application to the subtract input) is achieved in similar manner but with a shift in the opposite direction through gates B_0', B_1', B_2', etc.

The sign flipflop FF_S is set to the 0 state for pulses arriving at the add input and to the 1 state for pulses arriving at the subtract input. For example, if the counter is initially in the 0 state (FF_0 delivers pulses), upon application of a single pulse to the subtract line this pulse (through open gate B_2') passes through gate B_2'. This means that the state of the counter represents a negative number by its two's complement; i.e., with the two bits shown in the diagram this will be the code: $\boxed{1}10$ (two).

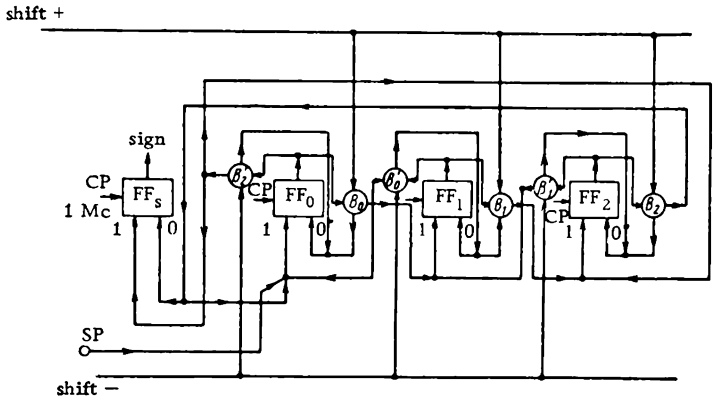


Fig. 144. Reversible counter.

Since each of the dynamic flipflops of a ring counter stores code 1 but once in a count cycle, it is not necessary that they have the high pulse repetition frequency required in a flipflop counter. Hence the frequency of count pulses may be increased while the maximum pulse repetition frequency for a flipflop may have a considerably lower value.

The schematic for one bit of a ring counter of 1 Mc (Fig. 145) consists of a high-power dynamic flipflop with storage capacitor and a gated amplifier. The dynamic flipflop consists of two stages: a low-power amplifier (P502A transistor) and a power amplifier (P601B transistor).

In the absence of voltage at capacitor C_4, diode D_2 is cut off by the bias voltage (8.7 v) applied to the "shift" line (*CPS*). The *CPS* pulses (with amplitude of 6.5 v) do not pass through the winding of transformer TR_3. In the presence of voltage at capacitor C_4 (when the flipflop is in the 1 state), the *CPS* pulses pass through the coincidence circuit and are amplified by transistor P16A, which triggers the succeeding flipflop of the counting ring.

The above described flipflop has been tested in a ring counter consisting of 18 stages; the count pulses have a frequency of one Mc and a pulse duration of 0.3-0.5 μsec. Operation of the circuit was not critical with respect to fluctuations in supply voltage. For example, the tolerances for pulse generator voltages and bias supplies were:

a) $E_{CP} = -3.5$ to -5.1 v;

$E_{CP(av)} = -4.3$ v;

$\Delta E_{CP} = \pm 0.8$ v or $\pm 11\%$;

b) $E_B = -2.2$ v to -5.8 v;

$E_{B(av)} = -4$ v;

$\Delta E_B = \pm 1$ v or $\pm 45\%$;

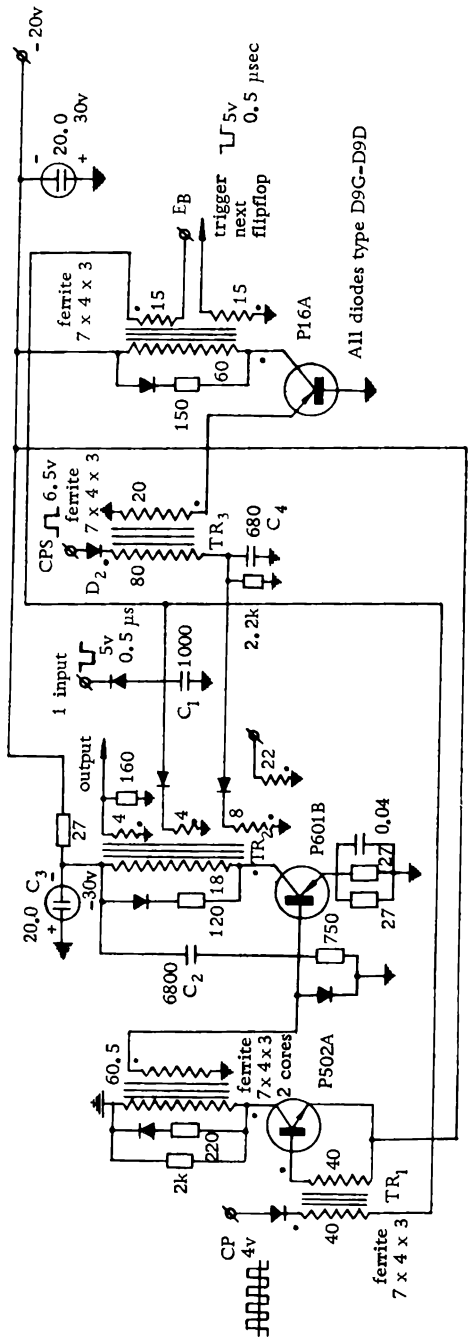


Fig. 145. One bit position of ring counter.

c) $E_{CPS} = -6.2$ v to -11.2 v;

$E_{CPS(av)} = -8.7$ v;

$\Delta E_{CPS} = \pm 2.5$ v or $\pm 30\%$.

The use of a two-stage amplifier in the dynamic flipflop provides considerably greater current gain than in a circuit with one transistor. A pulse power of the order of several watts (at a current of 1 a) is obtainable at the output of the P601B transistor; this is adequate for control of a large number of gates.

The first stage (P502A) amplifies the current passing through transformer TR_1 upon discharge of storage capacitor C_1. From the output of the stepdown transformer (turns ratio 12) the amplified signal is applied to the base-emitter circuit of power transistor P601B. Capacitor C_2 (6800 $\mu\mu$fd) in the collector-base circuit serves to improve the leading edge of the pulse at the output of P601B. Electrolytic capacitor C_3 and a resistor serve as a collector supply filter, eliminating the influence of noise arising from other flipflops of the counter. The flipflop feedback loop is closed from capacitor C_1 through the primary winding of transformer TR_1, diode D_1 and the CP pulse source. Auxiliary capacitor C_4 (680 $\mu\mu$fd) serves to store the pulse arriving from the flipflop output to the input of the gated amplifier, which consists of a diode-transformer coincidence circuit containing a diode, transformer TR_3 and amplifier transistor P16A.

15. DIODE-TRANSFORMER DECODERS

The usual diode-resistor decoding networks have the disadvantage that the output lines of the decoder have insufficient power to directly drive many logic circuits. A power amplifier is required at the output for each line, and a large number of output lines requires a large number of diodes.

The diode-transformer matrix described below gives a somewhat more economical decoder with output lines capable of driving many logic circuits without the use of intermediate amplifiers. Figure 146 shows such a decoder which consists of an array (or matrix) with horizontal and vertical lines fed either directly from the dynamic flipflops of two ring counters or by means of intermediate transistor amplifiers A. At each intersection of horizontal and vertical lines, a diode-transformer coincidence circuit is connected. A pulse appears at the output of the coincidence circuit if both lines (horizontal and vertical) to which a coincidence circuit is connected receive pulses from the dynamic flipflops of the ring counters. The diodes in the array are normally cut off by the bias voltage $+E_B$. For example, if current pulses from dynamic flipflops or from amplifiers A are applied to horizontal line x_1 and vertical line y_1, coincidence of these pulses in the transformer winding gives an output current pulse. If pulses from

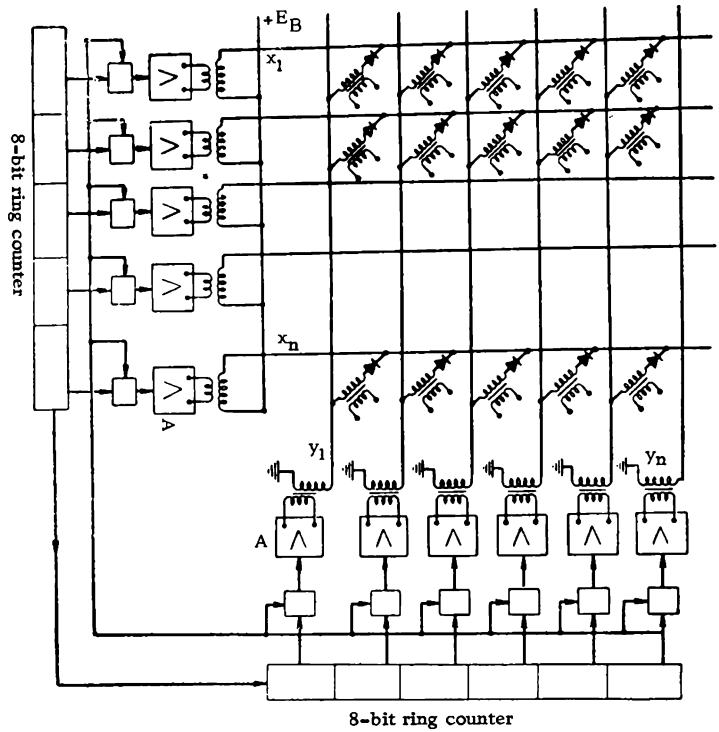


Fig. 146. Diode-transformer decoder.

the dynamic flipflop are fed to only one line x_1 or y_1, no current can pass through the transformer winding because the paths are blocked by the diodes. A diode-transformer matrix having n vertical lines and n horizontal lines permits n^2 output lines. Since each of the lines is fed by one dynamic flipflop, there are $2n$ dynamic flipflops. If input pulses are applied to both vertical and horizontal ring counters (as shown in Fig. 147), for each input pulse there is a corresponding one of the n^2 output lines of the diode-transformer matrix.

An extremely important property of the diode-transformer matrix is the possibility of obtaining a substantial power at its output lines for controlling many logic circuits. The output power of each of the transformers is determined by the power of the current pulse sources (dynamic flipflops or amplifiers) driving the horizontal and vertical lines as well as the efficiency of the diode-transformer coincidence circuit. In using ferrite cores for the transformers, the efficiency of the diode-transformer coincidence circuit is approximately 80-90%, depending upon the chosen number of turns, forward resistance of the diodes, and the CP pulse repetition frequency. If flipflops with CP frequency of 1 Mc are used as

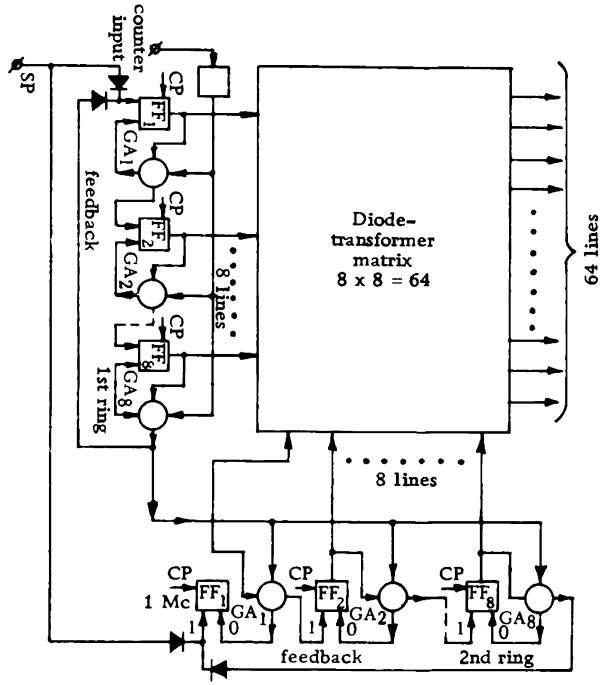


Fig. 147. Pulse-counter decoder.

the pulse sources driving the horizontal and vertical lines of the matrix, a power of the order of 0.5-1 w and higher can be obtained at each output line of the matrix.

In an experimental model of a diode-transformer matrix with 55 output lines (11 horizontal and 5 vertical), ferrite cores were used. The diodes were of type D2V. With load resistance of 500 ohms, each transformer of the matrix provided output pulses with amplitude of 20 v and duration of 0.3 μsec at a repetition frequency of 1 Mc.

The above described decoder with ring counters and diode-transformer matrix is designed for decoding the number of pulses arriving at the counter. However, the control circuits of a digital computer often require a decoder for a number code stored in the register. A decoder of this type for decoding a 6-bit number code is shown in Fig. 148.

The number code is stored by a six-bit register with dynamic flipflops. Three bits control the horizontal lines of the diode-transformer matrix and the other three bits control the vertical lines. Control is achieved as follows. Each 3-bit register is decoded by an ordinary diode decoder which gives eight output lines. By means of transistor amplifiers A_1-A_8, these lines set one of the dynamic flipflops FF_1-FF_8 to the 1 state. The dynamic flipflops directly control the lines of the diode-transformer matrix. The

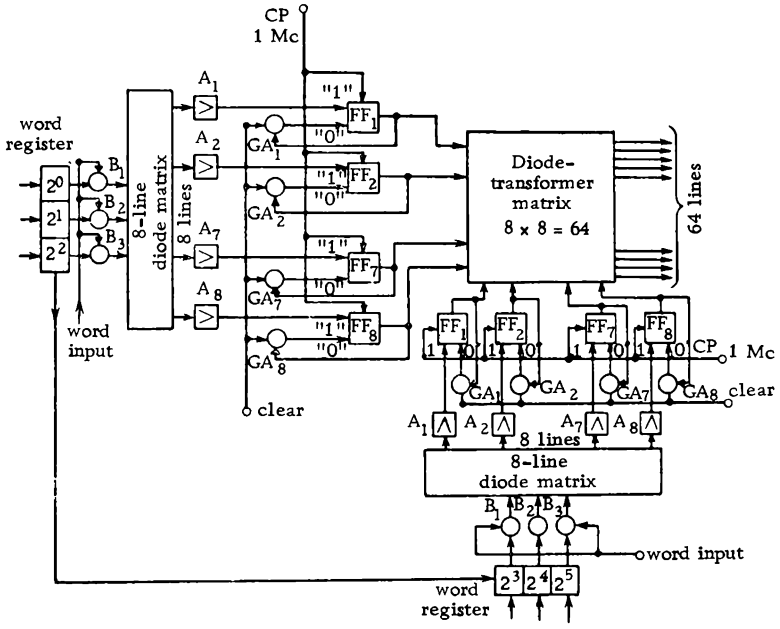


Fig. 148. Word decoder.

gated amplifiers GA_1-GA_8 serve to reset the flipflops to the 0 state. This operation occurs before a number code is inserted into the register. Gates B_1-B_3 connect the register to the decoder. At the output of the diode-transformer matrix (having 64 lines), there are obtained well shaped, high-power pulses with duration of 0.3 μsec at CP frequency of 1 Mc. A larger decoder can be constructed in a similar manner. It should be pointed out that a simple diode decoder with amplifier at each of the output lines (with the existing low-power high-frequency transistors) can not produce high-power output pulses with a frequency of 1Mc unless at least one vacuum tube is used at each of the output lines, whereas the above described decoder may be constructed with a pulse source having a total of two tubes.

16. SERIAL BINARY ADDERS

Serial single-bit adders operating at code summing frequency of 1 Mc are based on the basic AND, OR and NOT logic elements of the computer. Of the various versions of adders we discuss three circuits which may be constructed by the use of semiconductor diodes and transistors. The block diagram of a three-input adder using five OR circuits, three AND circuits and one NOT circuit is shown in Fig. 149.

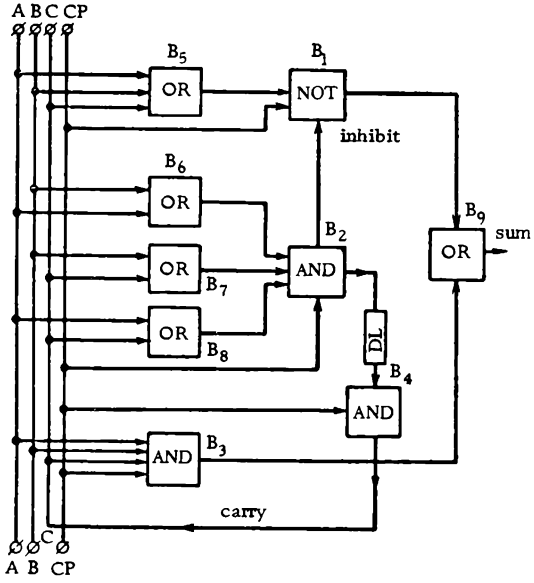


Fig. 149. Block diagram of three-input adder.

Lines A and B are inputs for the pulse codes of the two numbers to be added and line C for the pulses from one of the adder outputs delayed by one digit by delay line DL. Synchronization of the adder circuit is achieved by CP pulses or any other pulses from the control unit of the computer which coincide in phase with the CP pulses. If a pulse representing 1 arrives at one of lines A, B or C, the sum code is formed by gates B_5, B_1 and B_9.

Transfer of the pulse to the "sum" line is inhibited if pulses arrive simultaneously at lines A or B, B or C, A or C. In this case gate B_1 is cut off by means of the inhibit pulse from gate B_2 and the carry pulse from gate B_2 is transferred through delay line DL and gate B_4 to the carry line. Gate B_3 passes a pulse to the sum line when there is a coincidence of pulses at lines A, B and C and of the clock pulse CP. The diagram of a single-bit adder using diodes and transistors is shown in Fig. 150. The coincidence and anticoincidence circuit B_1 and coincidence circuits B_2 and B_3 are based on diode circuits with a common resistor R. The input pulses for the coincidence circuits block the corresponding diodes with negative pulses while the inhibit pulses are positive and unblock the diodes blocked by bias voltage $-E_B$.

The small bias voltage $+E_B$ in the inhibit gate serves to block the amplifier transistor in order to limit spurious signals. The OR circuits employ transformers with several windings and with diodes at each input. From each of the coincidence and anticoincidence circuits the output voltage is amplified by P402A transistors (diffused junction type) T_1-T_4. Coincidence circuit B_4 is based on a

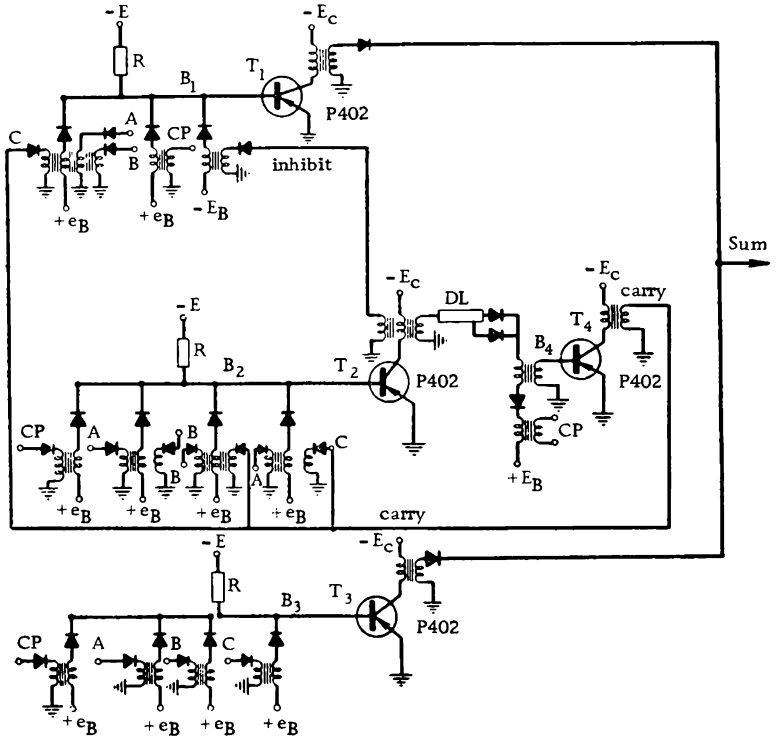


Fig. 150. Schematic of a single-bit adder.

diode-transformer arrangement, which is more suitable for amplification of the pulse arriving from the 1-μsec delay line DL. For widening of this pulse, delay line DL has two output branches with two buffer diodes. All diodes used in this circuit are D2V or D3V (with low forward resistance). The complete adder contains 15 ferrite-core transformers (with low forward resistance). The complete adder contains 15 ferrite-core transformers (including all input circuits), 27 diodes and 4 diffused-junction transistors of type P-402. The circuit is designed for operation with 0.3-μsec pulses with repetition frequency of 1 Mc.

Greater economy of components is obtained for the single-bit adder shown in Fig. 151. It consists of 4 coincidence circuits (AND gates), 2 anticoincidence circuits and 4 OR gates. In addition, there are four transistor amplifiers, transformers for signal inversion and a 1-μsec delay line for carry pulses. The input channels for number codes A and B, carry C and clock pulses CP must provide bipolar pulses; for this purpose, transformers with two output windings are used.

Addition is achieved by means of gates B_1-B_6 as follows:

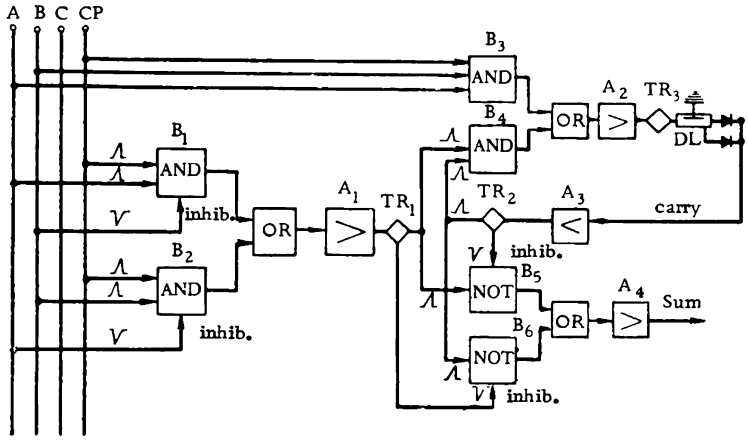


Fig. 151. Block diagram of a single-bit adder.

a) If there is a pulse at one input line and no pulse at the other input line, code A or B is transmitted to the "sum" output through gates B_1 (A, $\overline{B}$) or B_2 ($\overline{A}$, B).

b) If there is an input carry pulse C, the passage of pulses A or B to the "sum" output is inhibited by the carry pulse at gate B_5. In the same manner the passage of carry pulse C to the "sum" output is inhibited in the presence of an A or a B pulse at the input by gate B_6.

c) If A and B pulses exist simultaneously at the input, an output pulse is generated by B_3. A carry is also generated by gate B_4 in the presence of an input carry pulse C (stored in the delay line DL) and the pulse A or B.

A schematic of the adder is shown in Fig. 152. Coincidence gates B_1-B_3 with common resistor R have diodes which are cut off by pulses from input lines A and B and also have input carry C and CP lines. The output voltage of these gates is amplified by transistor amplifiers A_1-A_2. Anticoincidence gates B_5 and B_6 are diode-transformer circuits in which the diodes are cut off by inhibit pulses. The output voltage of gates B_5 and B_6 is amplified by transistor amplifier A_4. Transistor amplifier A_3 amplifies the pulses passing through delay line DL and delayed by 1 μsec for summing in the next bit. The adder circuit contains four transistors of type P402 and 18 semiconductor diodes of type D2V or D9D. In addition, there are ten ferrite-core transformers. It is necessary to ensure coincidence of the inhibit pulses in gates B_5 and B_6 with the pulses arriving at the second input.

A relatively simple adder requiring a small number of semiconductor elements is shown in Fig. 153. It consists of three coincidence circuits (B_1, B_2, B_3), two OR circuits and one inhibit circuit B_4.

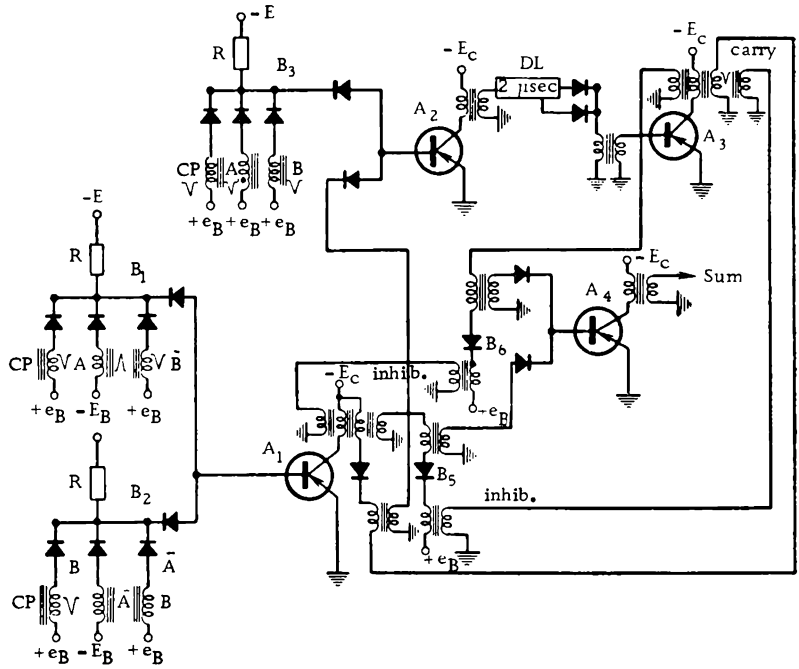


Fig. 152. Schematic of the adder in Fig. 151.

The "sum" output is here achieved through gate B_1 or gate B_4 and output carry achieved through gate B_2 or gate B_3. The schematic of this adder (Fig. 154) consists of 5 transistors and 13 diodes. The diode-transformer circuit with diode D_1 establishes coincidence of A and B input pulses. This circuit has two transformer windings at the output; one is used for the carry coincidence circuit (using transistor T_1) and the other is connected to the base of transistor

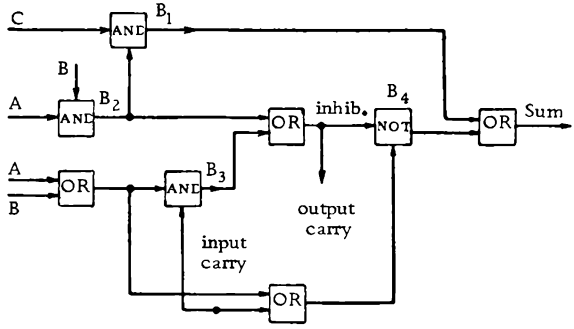


Fig. 153. Block diagram of a single-bit adder.

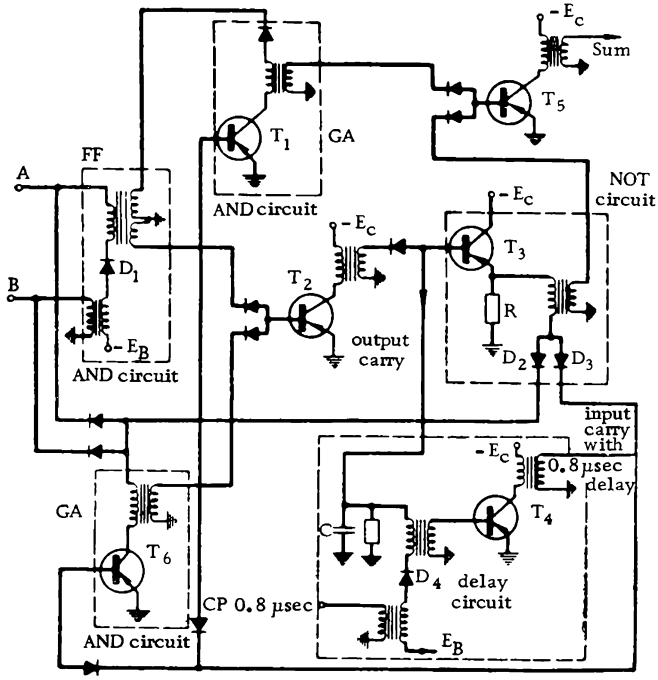


Fig. 154. Schematic for the adder in Fig. 153.

amplifier T_2. At the output of T_2 we obtain an output carry pulse. Transistor T_5 serves for amplification of the signal at the "sum" output. The emitter follower with T_3 is used as an inhibit circuit.

The carry pulse is applied to the base of transistor T_3, causing it to conduct. Diodes D_2 and D_3 are then cut off by the negative voltage drop at the emitter resistor R and no current flows through the winding of transformer TR_1. The output carry pulse is stored for approximately 0.8 μsec in the form of a charge at capacitor C. This capacitor is discharged by means of the diode-transformer coincidence circuit at the moment that diode D_4 is unblocked by a CPS pulse with a shift of 0.8 μsec relative to the clock pulses of the computer. This pulse is also generated by the power unit of the computer. The carry pulse thus delayed by 0.8 μsec is amplified by T_4 and applied to the bases of transistors T_1 and T_6 where it is stored briefly in the form of an accumulation of minority carriers in the transistor base. Upon arrival of a pulse at the collectors of T_1 and T_6 these transistors provide pulses at the coincidence circuit even in the case where these pulses are slightly out of coincidence in time (or in phase) with the previous-carry pulses. Thus, the adder is not critical with regard to pulse phases. The schematics shown in Figs. 152 and 154 are simplified versions.

Chapter IV

Storage Devices

1. MAGNETIC DRUM STORAGE

The magnetic drum storage is widely used in digital computers due to its simple operation, high storage capacity and permanent nature of the storage.

Numbers on a magnetic drum are stored in parallel or serial manner. In the parallel storage the number of tracks for writing a number corresponds to the number of bits. In the serial storage the bits of a number are recorded in sequence on a single track. Digital integrating computers most often use the serial method of storing numbers since it is the most economical method and requires the smallest number of read and write units. The capacity of the magnetic drum is proportional to the number of tracks and the packing density (i.e., the number of bits stored per centimeter of track length — 30-40 pulses per cm). The recording area may be increased by increasing the drum length and diameter. An increase in drum length increases the number of read-write tracks.

As the drum diameter is increased it becomes more difficult to maintain small spacing between the drum surface and heads; this spacing can be of the order of several tenths of a micron. Hence it is more practical to increase the storage capacity by increasing the packing density.

Reliable read-write operation and constant spacing are provided by using a "floating-head." In this method, high-pressure air between the read-write heads and the drum gives rise to a Bernoulli effect, preventing the heads from travelling away from the drum surface by "sucking" them back toward it. On the other hand, the air film existing between the heads and the drum surface prevents contact of the heads with the surface; the heads float, as it were, on an "air cushion."

In the serial read-write method, the number of integrators is determined by the track length. In order to increase the capcity of the integrating machine a serial-parallel method of recording numbers is used, wherein the numbers are coded, for example, in a binary-coded form.

The principal disadvantage of the magnetic drum storage lies in the use of mechanical parts moving at high speed, giving a

"gyroscopic effect." The read-write frequency stability depends on the constancy of drum speed, while the reliability of read-write operations depends on the precision with which the spacing between the drum and the heads is maintained.

The mechanical motion of the magnetic drum limits the pulse repetition frequency to a range of 100-200 kc, which in turn limits the operating speed of the DDA.

We shall discuss the read-write operations for a magnetic drum having a rotational speed of 3000-6000 rpm and higher.

The write operation is achieved by magnetization of a ferromagnetic coating on the drum surface 1 (Fig. 155). The write head 2 is a permalloy core with a gap and with a small winding (to decrease inductance); the drum-head spacing is 0.3-0.05 mm . A current pulse passing through the winding creates in the core gap *a* a field which magnetizes the coating on the drum. Each centimeter of drum circumference accomodates up to 40 pulses with a duration of 5-10 μsec. The recorded pulses are reproduced by means of read head 3; a voltage pulse is induced in winding 4 when the magnetized surface passes the gap.

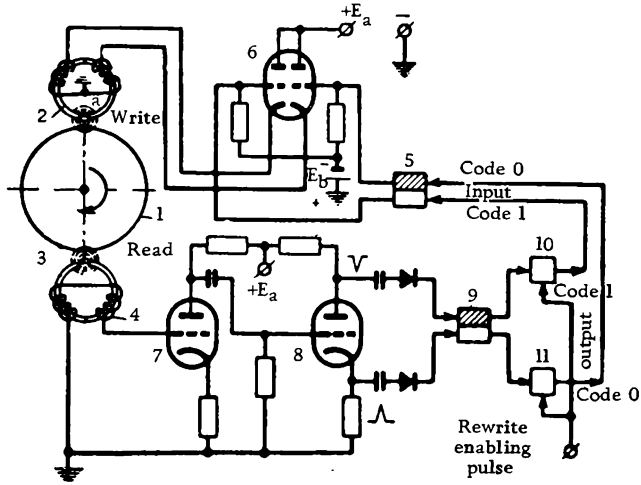


Fig. 155. Schematic of read-write operations on magnetic drum.

There are several methods of recording number codes on a magnetic drum. One of these is the two-level method. The surface of the magnetic drum is first magnetized by a current of one direction, which corresponds to code 0. In writing code 1 magnetization is achieved by a current of the opposite direction. For this purpose the write head has two windings with a grounded midpoint. The state of trigger flipflop 5 (code 0 or 1) allows cathode follower 6 to send the magnetizing current to one or the other winding of the write head.

Depending on the gap size and other head parameters, the emf induced in the read head is 0.05-0.1v. This emf is amplified by tube 7 and applied to phase inverter 8. The phase inverter output

gives two voltage pulses of opposite phase. Depending on the polarity of the induced emf, flipflop 9 is set to the 0 or 1 state. Flipflop 9 controls gates 10 and 11, which control the passage (to the output circuit) of the clock pulses recorded at equal intervals on a timing track on the magnetic drum. In writing new data on the drum it is not necessary to erase the old data since, if the pulse to be recorded is of sufficient amplitude, it magnetizes the surface to one or the other direction regardless of previous recording.

The codes written on the drum surface may be rewritten if the read head is connected with the write head in the manner shown in Fig. 155. The recorded code can be read and rewritten. Such recording is known as dynamic recording.

In some methods of magnetic recording, code 1 or 0 corresponds respectively to the presence or absence of a current pulse in the write head. The advantage of this method is that pulse recording may be achieved by using a-c amplifiers. However, magnetic recording using two levels permits the read-write frequency to be twice the recording frequency obtained by using pulses.

The length of a track between a read head and a write head becomes a delay line when the output of the read head is connected to the input of the write head as a feedback loop. The length of this delay line is determined by the track length between the write and read heads. The delay time τ is

$$\tau = n \cdot t_1,$$

where t_1 is the interval between two adjacent pulses and n is the number of pulses written on the track sector enclosed by the feedback.

In digital integrating computers the track length between the read and write heads is made as large as possible (for the y and S registers) in order to increase the number of integrators. An important advantage of using the magnetic drum storage for digital integrating computers is the fact that the latency (waiting time) in obtaining a number code from the memory is zero. In integration, the number codes arrive at the adder serially in the order of their recording on the magnetic drum and the use of a random-access storage device, as required in arithmetic machines, is not necessary.

2. READ-WRITE AMPLIFIERS FOR MAGNETIC-DRUM STORAGE

Nonreturn-to-zero recording on a magnetic drum requires that the transistor write amplifier provide a large current in the write head for a longer period of time. The transistor is constantly conducting but dissipates little power in this state. Nonreturn-to-zero recording is advantageous in that the pulse repetition frequency may be decreased by half and the packing density practically doubled; the latter, in turn, affords a relative decrease in drum dimensions.

The magnetic drum read amplifier for nonreturn-to-zero recording in Figure 156 consists of three stages. The read head winding is connected to the emitter circuit of transistor T_1 with grounded base. The high capacitance of capacitor C smooths the input signal in the base-emitter circuit. The bias voltage of +6 v in the emitter circuit serves to establish an adequate initial current level so that amplification of the signal occurs in the linear portion of the transistor characteristic. The initial current is regulated by resistor R, by which the zero voltage for the input stage of the amplifier is established.

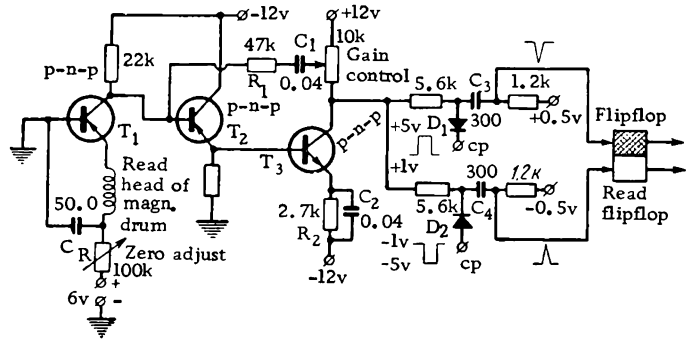


Fig. 156. Magnetic drum read amplifier.

The second stage, an emitter follower with transistor T_2, amplifies the signal power from the output of T_1. The output of the third stage gives an amplified voltage symmetrical with respect to ground. In the absence of signals, the collector voltage of T_3 (type n-p-n) is close to zero. Depending on the polarity of the signal in the read head, this voltage may be slightly positive or negative. The clock pulses CP cut off diodes D_1 and D_2, which, in the absence of CP, practically ground both positive and negative pulses from the output of T_3 and do not permit charging of capacitors C_3 or C_4. In the presence of the CP pulses, the output signal of transistor T_3 charges capacitor C_3 positively or C_4 negatively (depending on the signal polarity). At the termination of the CP, the charged capacitor, C_1 or C_2, is discharged through a 1.2-kilohm resistor, and differentiated voltage pulses of negative or positive polarity appear there to set the read flipflop to the state corresponding to the code to be read out. The R_2C_2 network in the emitter circuit of T_3 shortens the trailing edge of the pulse upon turn-off of the transistor and serves as feedback for stabilization of the operating point. The transient feedback through capacitor C_1 and resistor R_1 from the output of T_3 serves to improve the frequency response of the amplifier and to damp out oscillations. The output amplifier produces sufficiently high signal level to eliminate possible triggering of the flipflop by noise signals. The flipflop stores the read signal regardless of its phase relative to the clock pulses, since its storage is a result of charging either capacitor C_1 or C_2.

The magnetic drum write amplifier (Fig. 157) consists of a flipflop using n-p-n transistors (T_1 and T_2). Code 0 or 1 is applied to the 0 input or 1 input, and the flipflop is set to the corresponding state. The voltage taken from the collector of transistor T_1 or T_2 is amplified by a push-pull amplifier using transistors T_3 and T_4 or T_5 and T_6, respectively.

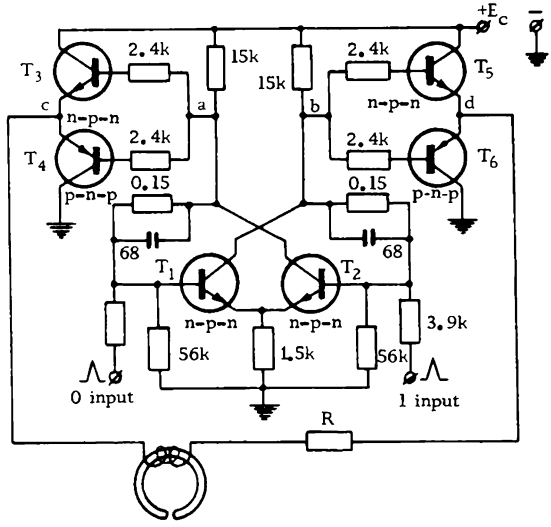


Fig. 157. Magnetic drum write amplifier.

The voltage polarity of the output amplifier between points c and d varies according to the state of the flipflop. This gives rise to a current in one or the other direction through winding L of the write head. Resistor R, connected in series with the head winding, serves to limit the write current. The amplifier is extremely stable in operation and relatively insensitive to changes in the parameter values of its components.

3. MAGNETIC INTERNAL MEMORY (MIM) WITH FERRITE PLATES OR CORES

Let us discuss a magnetic internal memory designed for a special-purpose digital integrating computer in serial operation. In the past, computers of this type used magnetic drum storages or ultrasonic delay lines (magnesium, quartz, magnetostrictive). The advantages of ferrite core memories are evident. They are compact, have short access time, provide nonvolatile storage and are extremely reliable in operation. Among the disadvantages limiting their use in special-purpose computers to date have been

the vacuum-tube control unit for the memory (drawing considerable power) and the complexity in making a ferrite array with a large number of cores.

In the MIM discussed here, semiconductor and ferrite elements are used for control circuits, and ferrite sheets with metal coating for storage elements. The control circuits developed by us for use as address-line switches make use of transistor amplifiers and diode-transformer matrices. The magnetic internal memory was developed for a digital integrating computer operating in the serial mode with a *CP* frequency of 1-2 Mc. The memory control unit consists of the following items: address-line switching circuit using a diode-transformer matrix and semiconductor power amplifiers; write register; write buffer register; write amplifiers for each of the bits; read amplifiers for each of the bits; read register.

4. MAGNETIC INTERNAL MEMORY

The magnetic internal memory (MIM) for a digital integrating computer, as shown in Fig. 158, consists of a magnetic memory proper, an intermediate register for storing the read-out number, and two intermediate registers for storing the number to be written into the memory.

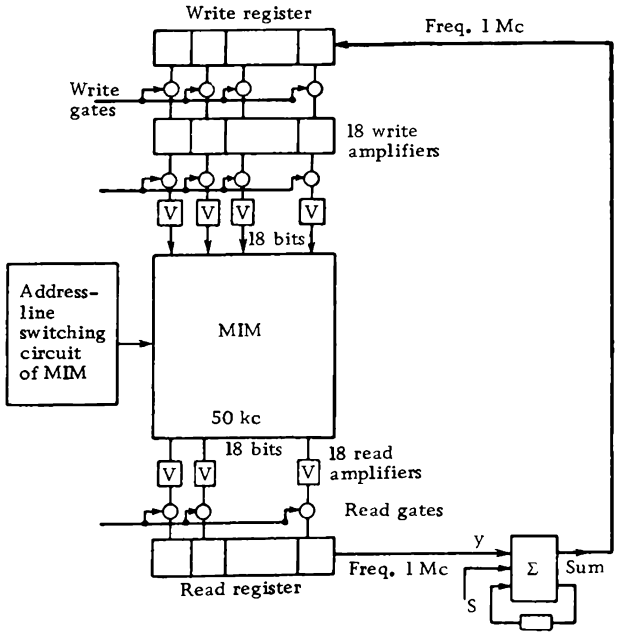


Fig. 158. Block diagram of MIM.

Write-in and read-out of the number codes in the memory proper are performed in parallel for all bit positions at a frequency of 50 kc. The number is read out from the memory proper to the read register, from which the number is serially transferred, bit by bit, to the adder at a frequency of 1 Mc. From the adder, the number is serially transferred to the write register at a frequency of 1 Mc. The write register may be eliminated if the read and write operations are performed from both ends in the read register. The number stored in the write register is again transferred in parallel for all bits to the memory proper at a frequency of 50 kc. Write-in and read-out of the numbers may be performed in time intervals especially allotted for this purpose; namely, read-out may be performed before the addition of the numbers and write-in at completion of the addition operation. However, at the expense of the memory access time, this increases the time interval required for the addition.

In order to decrease this time interval in the proposed memory, a brief interval is allocated at the beginning of the addition operation for read-out (of the order of 1-2 μsec). Write-in is then achieved during the addition operation by using two write registers. The number is transferred from the adder to the first register R_1. At the end of addition the number is transferred in parallel by the last pulse P_n from register R_1 to register R_2. At approximately the middle of the time interval of serial addition of the next number, the previous number is transferred from register R_2 to the memory proper for write-in.

For example, if the cycle time for addition of two 18-bit numbers is 20 μsec, read-out of the number from the memory proper is achieved in 2 μsec (before the beginning of addition) and the remaining 18 μsec are devoted to serial addition of the 18 bits (including the last pulse P_n). During read-out, the total current is applied to the address lines for magnetic switching. Writing the number from register R_2 into the memory proper is performed approximately at the middle of the addition operation cycle. To write, a write current equal to half the current required for magnetic switching is applied for 2 μsec to the address line. At the same time, a current equal to half (or one third) the write current with twice the duration (4 μsec) is applied to the write winding in order to ensure coincidence of the two half-select currents. To write code 1 a current of one polarity is applied to the write winding, and to write code 0 a current of opposite polarity is applied.

During each memory cycle (20 μsec) one read operation and one write operation are performed. For example, read-out of the contents of integrator 2 is followed by write-in of a number into integrator 1; read-out of the contents of integrator 3 is followed by write-in of a number into in integrator 2, etc, as shown in the time diagram of Fig. 159. In order to achieve such a write operation, the number code is delayed by 10 μsec before its application from the write register. For this purpose it is necessary to have two write registers. The first register stores the number arriving

from the adder. Upon conclusion of addition, the number is transferred in parallel from the first register to the second register. A write signal then initiates the transfer of the number code to the memory proper at 10 μsec after read-out of the next code. This permits the read and write operations to be achieved with longer memory cycle time without increasing the time for serial addition.

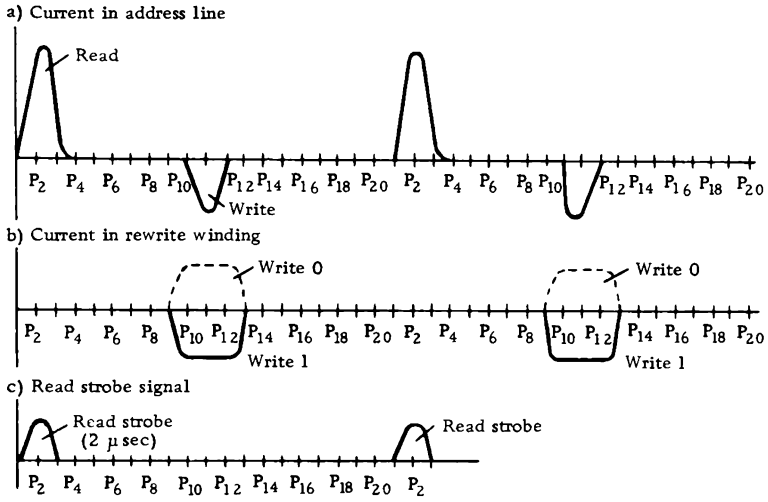

a) Current in address line
Read
P2 P4 P6 P8 P10 P12 P14 P16 P18 P20 P2 P4 P6 P8 P10 P12 P14 P16 P18 P20
Write
b) Current in rewrite winding
Write 0
Write 0
P2 P4 P6 P8 P10 P12 P14 P16 P18 P20 P2 P4 P6 P8 P10 P12 P14 P16 P18 P20
Write 1
Write 1
c) Read strobe signal
Read strobe
(2 μsec)
Read strobe
P2 P4 P6 P8 P10 P12 P14 P16 P18 P20 P2

Fig. 159. Time diagram of read-write cycle.

A ferrite plate with 256 apertures (each aperture is equivalent to a core) is used as the memory proper. On this plate there are projecting ribs which separate the rows of apertures in the horizontal and vertical directions (Fig. 160). The horizontal ribs on the front of the plate are displaced relative to those on the back of the plate by one aperture. The surface of the plate (front and back), including the aperture walls is coated with a metal layer. The metal coating is not applied to the ribs and the sides 1 and 3 of the plate. Two current conducting leads are connected to sides 2 and 4. Due to the relative displacement of the ribs separating the aperture rows on both sides of the plate, the metal coating forms a conductor threading all 256 apertures in series.

Through each aperture of the ferrite plate there pass two conductors: one conductor (formed by the metal coating) serves as the read-write winding; the other conductor (passing through all corresponding apertures in the stack of plates) serves as the address line. The number of address lines is equal to the number of apertures (256). The apertures along the address line are cores which store all the bits of a number. Two corresponding apertures of a pair of plates store one bit. One pair of plates stores the codes for all corresponding bits of 256 numbers. The write and read

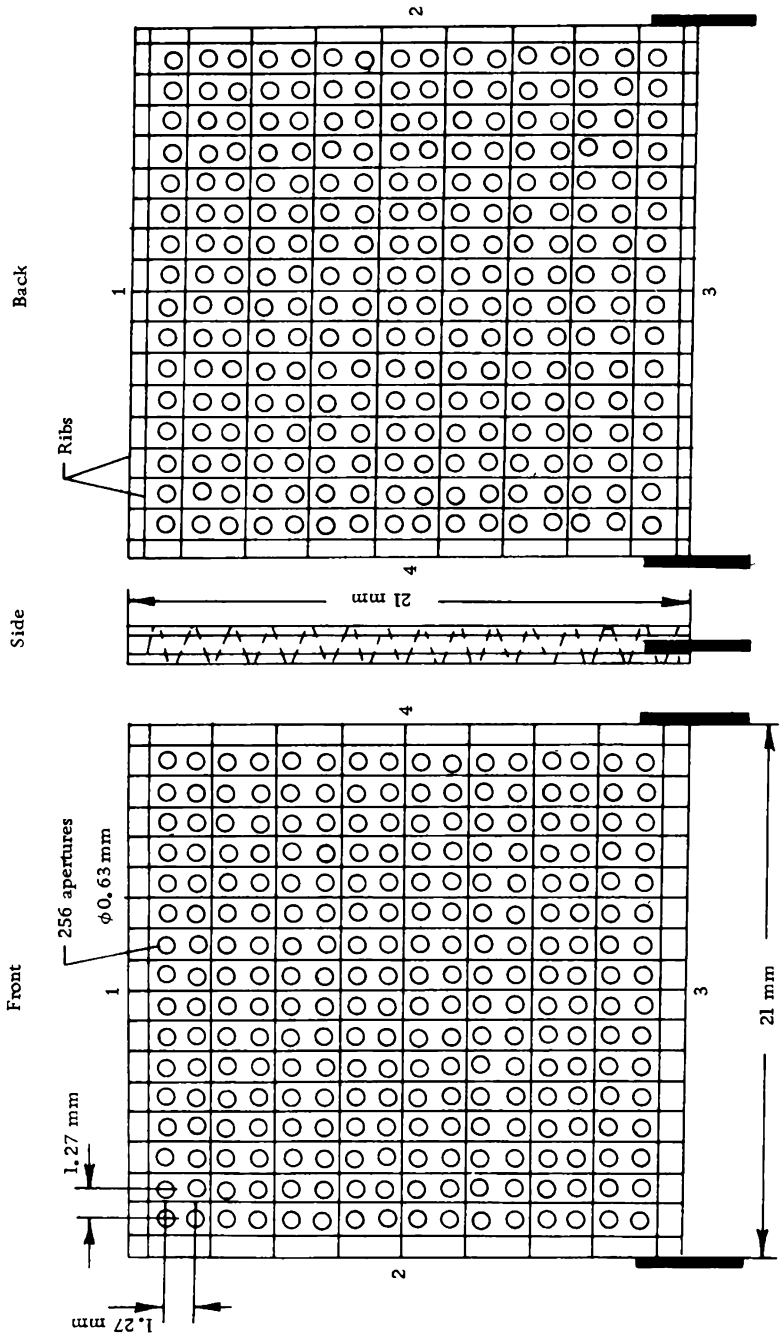


Fig. 160. Arrangement of apertures and ribs on ferrite plate.

windings formed by the metal coating of a pair of plates are connected in series in such a manner that the magnetizing currents for each pair of corresponding apertures are of opposite direction.

Fig. 161 shows the diagram for the read and write windings of one pair of ferrite plates storing all corresponding bits. Instead of one address line for both read and write operations, through each aperture are threaded two insulated address lines, one of which is used for the read and the other for the write operation. Thus, the read and write operations may be performed in the different time intervals corresponding to two adjacent integrators. The read operation is first performed and then the write operation. In read-out of the single bit stored by a pair of apertures in two adjacent ferrite plates (Fig. 162a), the full current is applied to the read address line and magnetizes both cores with the flux Φ_r in the clockwise direction.

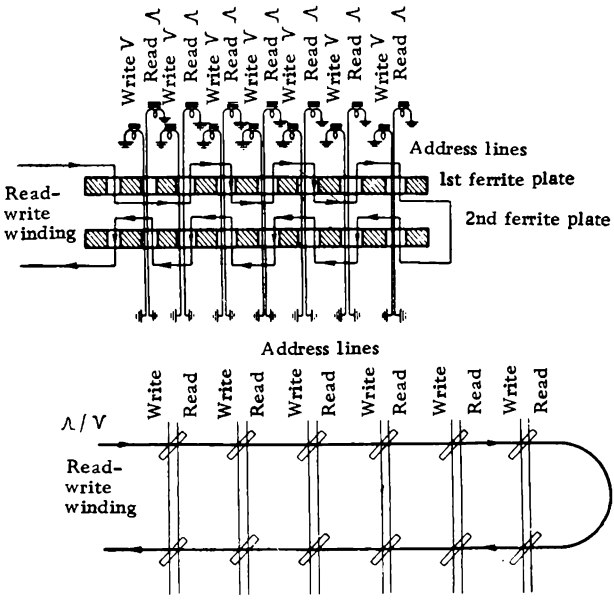


Fig. 161. Read and write lines for one pair of ferrite plates.

To write, a half current is applied to the write address line and, at the same time, a half current to the write winding. To write 1 (Fig. 162b), the current in the write winding has a direction magnetizing the ferrite around one aperture in one direction (clockwise) and around the other aperture in the other direction (counter-clockwise). As a result, the fluxes generated by the current in the address line and in the write winding subtract around aperture 1 and add around aperture 2. Flux reversal occurs only around aperture 2, since the fluxes cancel out around aperture 1. To write 0 (Fig. 162c), the direction of current in the write winding

is opposite to that in writing 1. In this case, the fluxes add around aperture 1 and subtract around aperture 2, so that flux reversal occurs only at aperture 1.

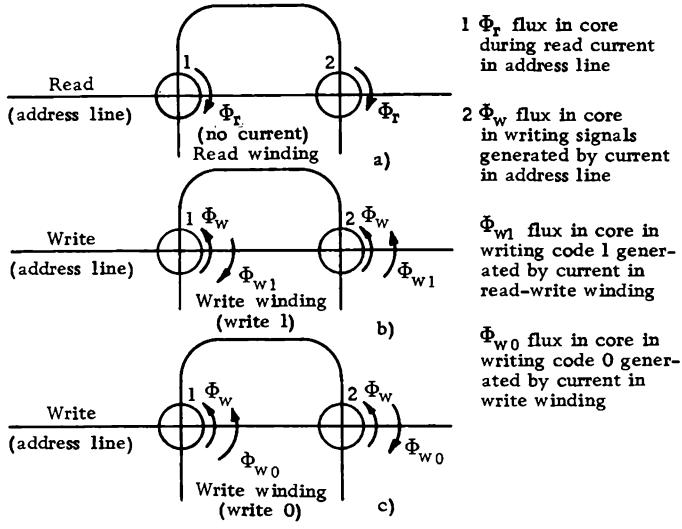


Fig. 162. Summing of magnetic fluxes in write and read operations.

To write 1, aperture 2 stores the state representing code 1 and in writing code 0, aperture 1 assumes the state representing code 0. Hence, the read signal will have different polarity for code 1 and 0. This is of particular significance, since it permits qualitative discrimination of small signals according to their polarity for codes 1 and 0 even in the presence of noise. The presence of two plates instead of one for each bit (Fig. 163) also offers a constant impedance for the address line during the write and read operations regardless of the number code.

Upon passage of the full read current through the address line all pairs of apertures are set to zero. In writing a number of 1's and 0's, the number of apertures in the address line whose states represent code 1 is equal to the number of apertures whose states represent code 0. This means that the load in the address line and the load in the write winding are always the same.

The use of ferrite plates instead of cores requires less current in the address lines and in the write winding. For example, for a 21 × 21-mm plate with 16 × 16 = 256 apertures (diameter 0.63 mm), the full current in the address line was 500 ma for 5 μsec and the current in the write winding was 100-150 ma. The write current in the write winding began 3 μsec after application of current to the address line. During read-out the signal in the read winding was only 20 mv for about 1.7 μsec. During this operation, the noise

from the remaining apertures during rewrite may exceed 1 v. Hence during rewrite, the read amplifier must be cut off by a special pulse. In order to prevent flux interference among adjacent apertures, the current must not exceed 500 ma. The advantage of the above described MIM lies in the fact that write-in and read-out are separated by a relatively large interval of 9 μ sec, so that the noise level at read-out is at a negligible value. The use of two apertures on two ferrite plates for storing a single bit eliminates the noise occurring in read-out of code 0 due to the flat portion of the hysteresis loop not being horizontal (or squareness ratio not being one). This results in lower homogeneity requirements for the ferrite plate than for ferrite cores.

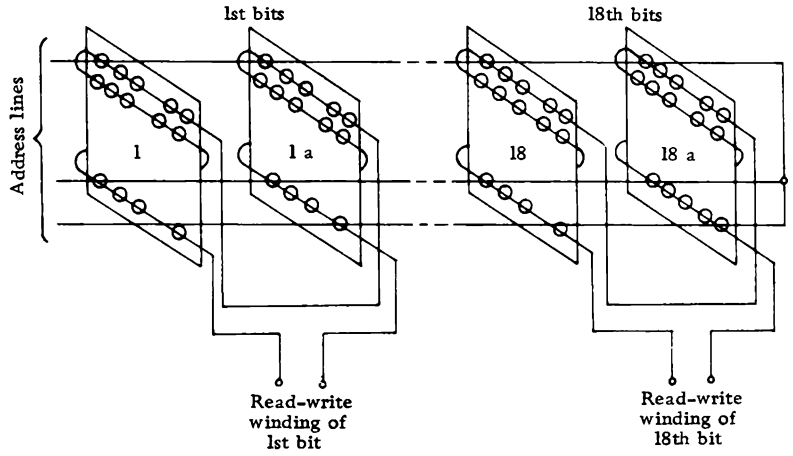


Fig. 163. Wiring diagram for ferrite plates of MIM.

Another feature of the above described MIM system is the possibility of "forced" read-out, reducing the memory access time to a few microseconds. In the MIM control unit described below, two disadvantages of the MIM using a core matrix switch are eliminated by using a diode-transformer matrix. The mentioned disadvantages of using a core matrix switch are the noise in the unselected address lines and the need of a large d-c bias which draws considerable power.

5. ADDRESS DECODING

Existing address decoding circuits of a magnetic internal memory using a ferrite core matrix switch require the use of a large number of vacuum tubes and draw considerable power; this leads to a large, complex memory control unit.

The author has proposed a memory control unit in which address-line selection is achieved by means of a diode-transformer matrix (Fig. 164) driven by transistorized pulse amplifiers.

Count of the integrator numbers is achieved by a ring counter with dynamic flipflops connected in series according to the sequence of the integrators. Each counter stage controls a transistor power amplifier. Since the *CP* pulses to the dynamic flipflops have a repetition frequency of 1 Mc and a duration of 0.3 μ sec, control of the power amplifier requires that the control pulse be stretched to 1 μ sec; this is achieved by an ordinary pulse stretcher consisting of a delay line and buffer diodes (not shown in the figure).

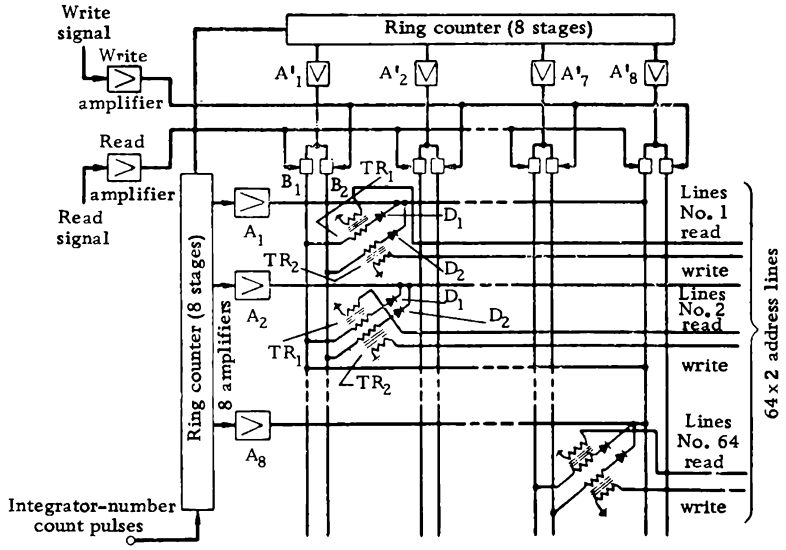


Fig. 164. MIM diode-transformer matrix.

The amplifiers drive the horizontal and vertical lines of the diode-transformer matrix. For a matrix having, for example, 64 address lines at the output, it is necessary to have 8 horizontal and 8 vertical drive amplifiers. Each intersection in the diode-transformer matrix is a coincidence circuit. It consists of two transformers TR_1 and TR_2 and diodes D_1 and D_2, which are normally cut off by the bias voltage. Upon coincidence of voltage pulses at one horizontal and one vertical line, a current pulse is induced in the transformer winding located at the intersection of these lines. The output winding of each transformer drives one of the 64 address lines of the diode-transformer matrix.

In order to obtain a bipolar signal during the write and read operations, each intersection of the diode-transformer matrix consists of two transformers connected to the vertical lines of the matrix. The number of these lines is twice that required in an ordinary matrix. The currents flowing through the primary winding of transformer TR_1 or TR_2 induce in the secondary a voltage of opposite polarity, which drives the address lines. Selection of any

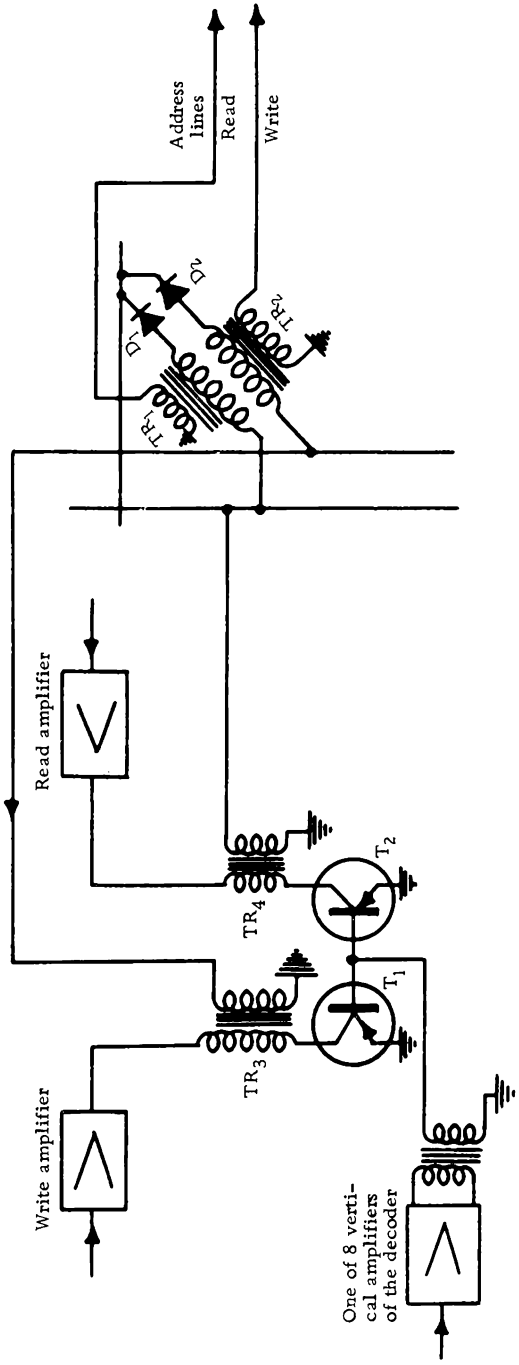


Fig. 165. Element of a MIM diode-transformer matrix.

one address line during write or read operation is achieved through gates B_1 or B_2, which is driven to conduct by an amplifier connected to one of the 16 vertical lines of the matrix. Figure 165 shows the circuit of an element of a diode-transformer matrix which will serve to explain the above described principle of selection. The gates here are transistors T_1 and T_2, whose collectors are fed by the write or read amplifier. Transistors T_1 and T_2 are enabled to conduct by a voltage pulse from one of the 8 vertical amplifiers of the diode-transformer matrix. The secondary windings of transformers TR_3 or TR_4 feed the vertical lines of the diode-transformer matrix with transformers TR_1 or TR_2. The matrix in turn transmits the write or read pulse to one of the address lines.

In using transistors in a coincidence circuit it is not necessary to achieve precise phase coincidence of the pulses from the decoder amplifier and from the read or write amplifier. The pulse from the decoder amplifier, arriving previously, is "stored" in the form of an accumulation of "holes" in the semiconductor, which retains its conductivity for a short while after disappearance of the pulse in the base-emitter circuit. Figure 166 shows the diagram of the complete unit for control of a magnetic memory consisting of 256 numbers (integrators) and using a diode-transformer matrix. The diode-transformer matrix output has 64 write address lines and 64 read address lines. The 256 apertures in the ferrite plate are divided into four groups of 64 apertures each. Each of the 64 address lines passes in parallel through one of the apertures in each group. The ends of all 64 write address lines and read address lines of each group are joined and connected to a grounded current source by means of transistor gates $T_1 - T_1'$ to $T_4 - T_4'$. Each of these transistors is controlled by one of the four stages of a group number counter, which is switched by pulses arriving from the output of the integrator number counter. After counting off all 64 integrator numbers, the group number is switched. The 64 address lines of the next group are connected in sequence. Division of the address lines into four groups permits a considerable decrease in the number of semiconductor and magnetic elements in the diode-transformer matrix, which, instead of 256 address lines, has only 64 lines (i.e., one-fourth as many elements).

The read-write winding for each pair of ferrite plates passes in sequence through all 512 apertures of the two plates. The number of such windings is equal to the number of bits in a word (for example, 18). If the windings are used for read-out of numbers from the memory proper, they deliver the read signals to 18 read amplifiers. The read amplifiers must amplify read signals of different polarities, since code 1 and code 0 have pulses of different polarities. At the amplifier output a pulse corresponding to code 1 causes the last stage to conduct, while a pulse corresponding to code 0 cuts it off. Through gates $B_1 - B_{18}$ the amplified read signals pass to the read shift register, from which the number code is serially transferred at a frequency of 1 Mc to the DDA. The result of addition is serially transmitted to write register

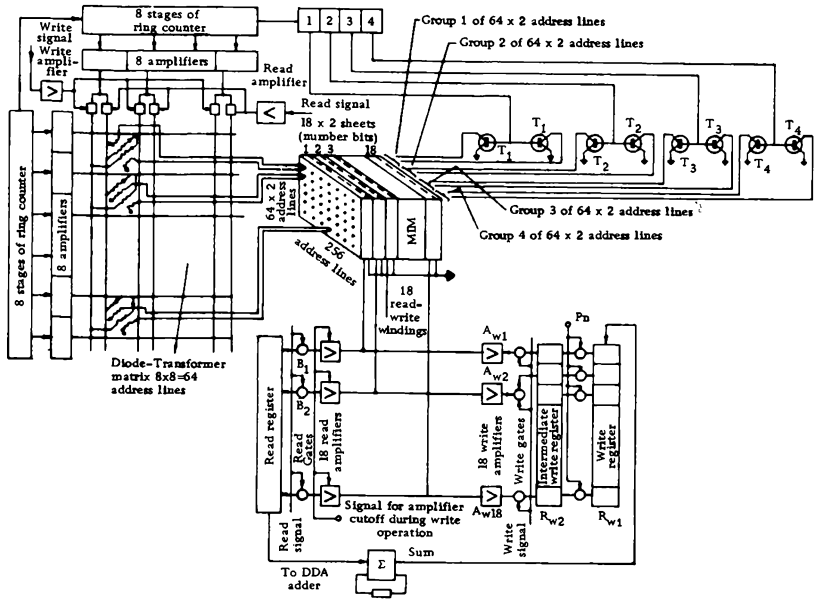


Fig. 166. Complete address decoding diagram.

R_{W1}. The number code from register R_{W1} is transferred by the last pulse P_n in parallel to the intermediate write register R_{W2}, from which, upon the write signal, it is transferred to the input of write amplifiers A_{W1}—A_{W18}. In writing code 1 the transistor write amplifiers deliver a pulse of positive polarity and in writing code 1 a pulse of negative polarity. For this purpose the write amplifier has a push-pull circuit with two p-n-p transistors and two n-p-n transistors. The pulses are transmitted directly for the write register flipflops storing code 1 to corresponding amplifier inputs and the pulses are cut off for those flipflops storing code 0. The pulses at the write amplifier output must have a magnitude of about 100-200 ma at a voltage of 1-2 v with leading and trailing edge durations of 1 μ sec.

6. READ AND WRITE OPERATIONS

In a magnetic internal memory using ferrite cores it is possible to select a number code or word for read or write by actuating one address line. Such selection is called linear selection. A memory using linear selection is two-dimensional and differs from a three-dimensional memory which employs coincident current selection.

The diagram of one memory matrix or array is shown in Fig. 167. The matrix has 16 vertical columns and 64 horizontal rows, giving a capacity of 64 words with 16 bits per word (15 number

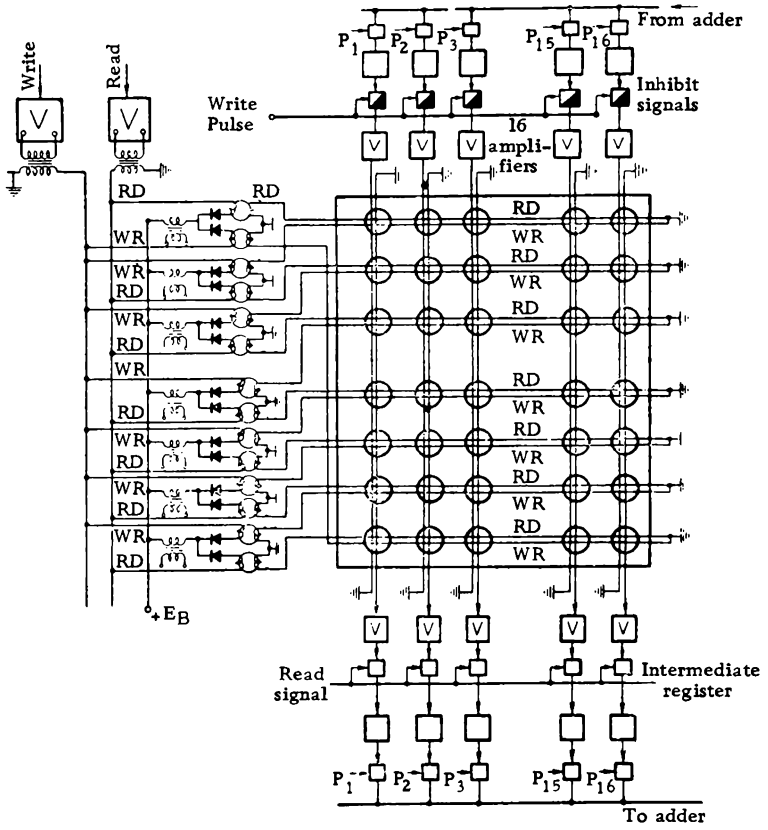


Fig. 167. MIM matrix.

bits plus a sign bit). Each vertical column of the matrix consists of corresponding bits of all words. A single matrix contains (16 × 64 =) 1024 cores. A single channel of computation in a digital integrating computer with a capacity of 64 integrators requires two matrices with 1024 cores each. An increase in memory capacity is achieved either by increasing the number of matrices (e.g., increasing parallel channels of computation in the computer) or by increasing the number of rows in each matrix. For example, for a digital integrating computer with two channels it is necessary to have four matrices for a total capcity of 64 × 2 = 128 integrators.

In Fig. 167, there are four conductors threaded through each ferrite core of the matrix. One horizontal conductor, passing through 16 cores, sets the cores to the 0 state during read-out and the other horizontal conductor sets the cores to the 1 state during the write operation. One vertical conductor serves for application of the inhibit pulse during the write operation, while

the other vertical conductor is the output winding for read-out. The horizontal conductors thus serve as address lines, while the vertical conductors are used for the read-write operations.

During read-out, a negative pulse is applied to a selected horizontal address line and sets all cores of this row to the 0 state. Flux reversal will occur for those cores in the 1 state. In the vertical output windings passing through these cores pulses are induced as read-out signals.

The read-out pulse for each bit is amplified by a transistor amplifier and the number is stored in the read register. Transfer of the read-out signals to the register is achieved in a bit-parallel fashion. The word in the read register is transferred bit by bit to the adder of the integrating unit by the pulses from lines P_1—P_{16}. In writing a word, a positive pulse is applied to a selected horizontal address line and sets all cores of this line to the 1 state except those that are inhibited.

Both the above-mentioned positive (during the write operation) and negative (during read) pulses applied to a selected address line are derived from one of 64 pairs of ferrite-core transformers which are associated with the 64 address lines. The primary winding of one transformer in a pair is fed from the read amplifier A_r and that of the other from the write amplifier A_w. The windings are so connected that the pulse polarities in the address line during the read and write operations are opposite. Current in the primary windings is controlled by an enabling signal in the diode-transformer matrix. In the presence of this enabling write or read signal, one of 64 pairs of diodes cut off by the bias voltage is unblocked and current flows through the primary winding. Each pair of transformers and its associated diodes and bias source E_B form a diode-transformer coincidence circuit passing a pulse to the write or read address line upon coincidence of two pulses: a pulse from a read or write amplifier and a pulse from one of the integrator lines (1, 2, 3, etc) from the control unit of the computer. Pulses from the integrator lines, applied in time sequence to the transformers, generate in the secondary windings a voltage compensating the bias voltage $+E_B$; as a result, the 64 pairs of diodes conduct current in sequence.

The word to be written into the memory arrives serially, bit by bit, from the adder of the integrating unit at a frequency of 1 Mc. By means of 16 gates controlled by the bit lines of the computer control unit, the serial word is stored in the write register. Each bit position of this write register controls an anticoincidence gate which, in the presence of a write signal, produces a signal corresponding to the complement of the state of the bit position. These complemented signals are amplified by transistor power amplifiers into half-select currents which drive the vertical lines to inhibit establishment of the 1 state in those bit positions whose code in the register is 0. The inhibition is due to the subtraction of a half-select current from the full current applied to the horizontal line during the write operation. If a bit position in the write register is 1, the negative half-select

pulse is not generated in the vertical inhibit line and the write pulse in the horizontal (address) line switches the core to the 1 state.

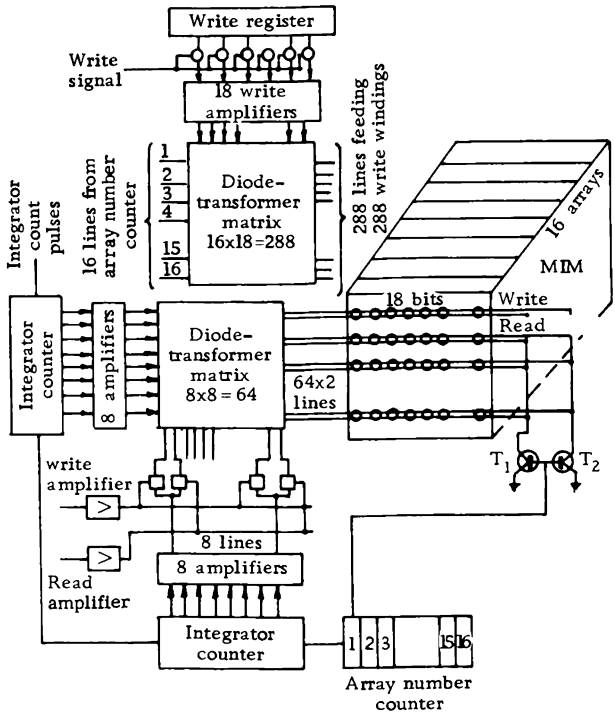


Fig. 168. Block diagram of a MIM.

During application of the inhibit pulse to a vertical line, the ferrite cores on the vertical line are not switched (with the exception of the core on the selected address line). This means that the total resistance of the inhibit line remains almost the same and little power is required from the write amplifiers.

The read and write operations of the above MIM produce little spurious signal in the read winding during read-out. During the write operation in the unselected cores lying on the vertical lines (through which the half-select inhibit currents pass), the half-select currents can not switch the core state. During read-out, the transients and spurious signals arising as a result of the write operation have no significant effect on the read operation because the read interval is displaced in time relative to the write interval and because the noise in the read winding is cancelled by threading half of the cores of the vertical column in one direction and the other half in the other direction. The read-out signal in the output winding is bipolar (i.e., positive or negative). Hence the read amplifier must be capable of rectifying pulses of either polarity.

For storage of a large number of words, the MIM requires several ferrite-core matrices or arrays. Fig. 168 shows the block diagram of a MIM with 1024 words. There are 16 ferrite-core arrays, each array having 64 18-bit words. The write and read address lines are selected by a diode-transformer matrix having 8 horizontal and 8 vertical lines.

The 64×2 output lines of the diode-transformer matrix are connected in parallel to the address lines of the 16 arrays. The current in an address line is controlled by transistor gates T_2 (during the write operation) or T_1 (during read-out). Through these transistors flow the currents from all write address lines or read address lines of one array. The number of pairs of transistors T_1 and T_2 is equal to the number of arrays (there are 16 pairs in Fig. 168). Transistor pairs T_1 and T_2 are controlled by an array number counter which counts the output pulses of the integrator number counter; each count of the former counter represents counting of 64 integrators.

In order that the half-select write current will pass through the cores of only one array, connection of the write winding of each bit position of an array to the corresponding write amplifier is controlled by a diode-transformer matrix. This diode-transformer matrix has 16 horizontal lines for the 16 arrays and 18 vertical lines for the 18 write amplifiers. The output of the diode-transformer matrix consists of 288 lines. The array number controls the connections of the 18 write amplifiers to the 18 write windings of the selected array.

The use of 288 rather than 18 write windings shortens the circuit path for the half-write current by a factor of 16 (i.e., from 288 to 18 cores). This results in independent operation of each array and in increased operational reliability of the memory.

7. DRIVERS

The use of power transistors (type P601) permits construction of simple and reliable drivers and amplifiers without use of intermediate transformers to drive the address lines (or word lines). Figure 169 shows the schematic of a switch for selecting address lines. In this diagram, the write and read address lines Z_{lw} and Z_{lr} of the memory are connected at one end to the emitter-follower output of transistor T_1, which is controlled by the address line (integrator number) select signal. The other ends of the address lines are connected to the collectors of transistors T_2 and T_3, which are controlled by the write or read signals for each plate of the memory. The current amplifier using transistor T_1 will be referred to as the address current driver, while the amplifiers using transistors T_2 and T_3 will be called the write and read address drivers. This address line selection switch was developed for a MIM having four plates, each plate with 16 word lines.

The signal at input X of the address current driver drives transistor T_1 to saturation. The voltage at point 1 becomes almost

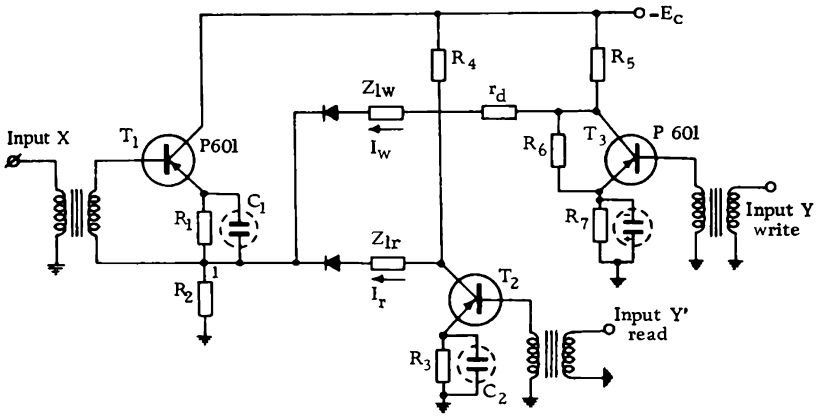


Fig. 169. Address line selection switch.

equal to E_c since resistor T_2 has a small value while capacitor T_3 has a large value (of the order of 80 μfd). With simultaneous application of a pulse to input Y, transistor T_2 is also driven to saturation. The collector voltage of R_1 drops almost to zero and a large current, sufficient to switch the cores, flows through the load circuit Z_{lr}. After a certain time interval, a write control signal is again simultaneously applied to input X and to input Y. Transistors T_1 and C_1 are again in saturation. An address current pulse now flows in the write address line; the current pulse amplitude can be set within certain limits by resistor r_d. With the arrival of pulses at input X alone or at inputs Y and Y', no current pulses arise in the read or write address line. The noise occurring in these lines is negligible.

The schematic of an address current driver is shown in Fig. 170. Transistor T_1 (type P16A) is a gated pulse amplifier. Capacitor C_1 is charged by one of the integrator pulse "trains" P_1-P_{16} arriving at input *In* 1 from the counting circuits for the 16 integrators. Input *In* 2 is for a single write pulse and input *In* 4 for a single read pulse. At the end of the "train," capacitor C_1 is discharged by the last bit number pulse P_n. Resistor R_5 serves to shunt the reverse resistance of diode D_1 and to discharge the capacitor whose potential is formed by the divider consisting of the reverse resistances of diodes D_1-D_3.

Transistor T_2 (also of type P16A) is operated as an emitter follower. It serves to match the somewhat high output resistance of amplifier T_1 with the low input resistance of power transistor T_3 (type P601).

The schematic of a read address driver is shown in Fig. 171. Transistor T_1 is operated as a gated pulse amplifier and transistor T_2 as an emitter follower matching the output resistance of T_1 with the input resistance of T_3. Capacitor C_1 is charged by one of the pulse "trains" of the integrator group numbers G_1-G_4. Through

input *In* 3 and the amplifier pass 16 read pulses (according to the number of integrators in the group). At the end of the "train," capacitor C_1 is discharged by discharge pulse P_{16}. Resistor R_1 discharges capacitor C_1, whose potential is formed by the reverse-resistance divider of diodes D_1 and D_2.

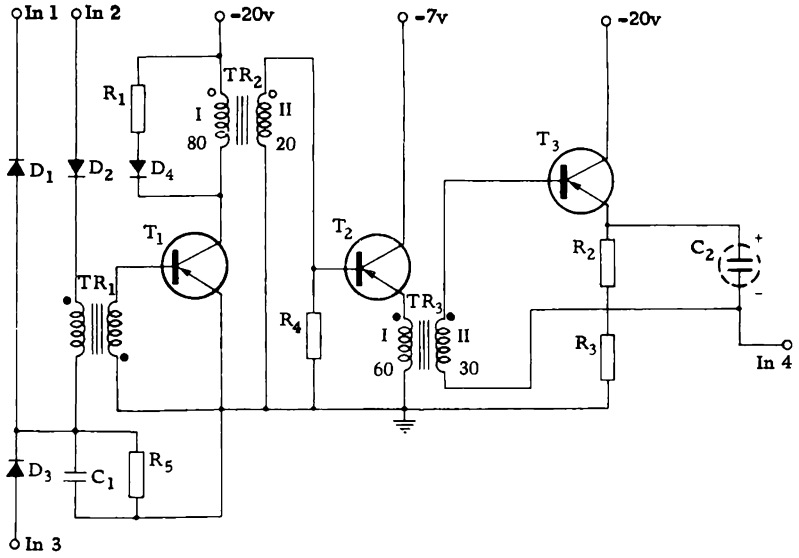


Fig. 170. Schematic of an address current driver.

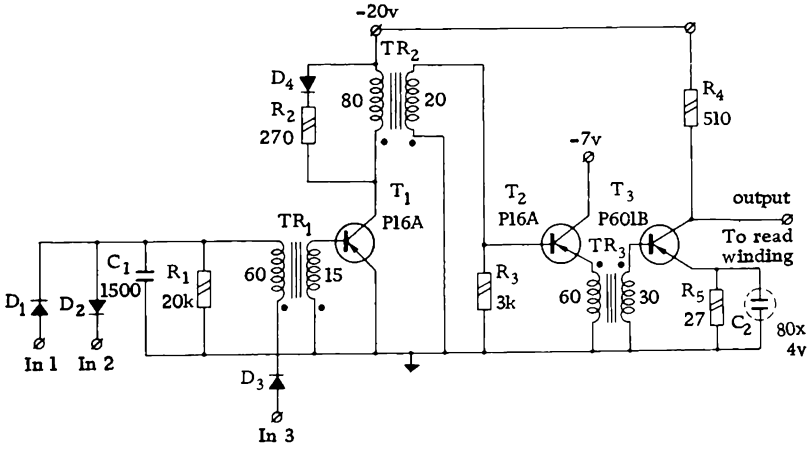


Fig. 171. Schematic of a read address driver.

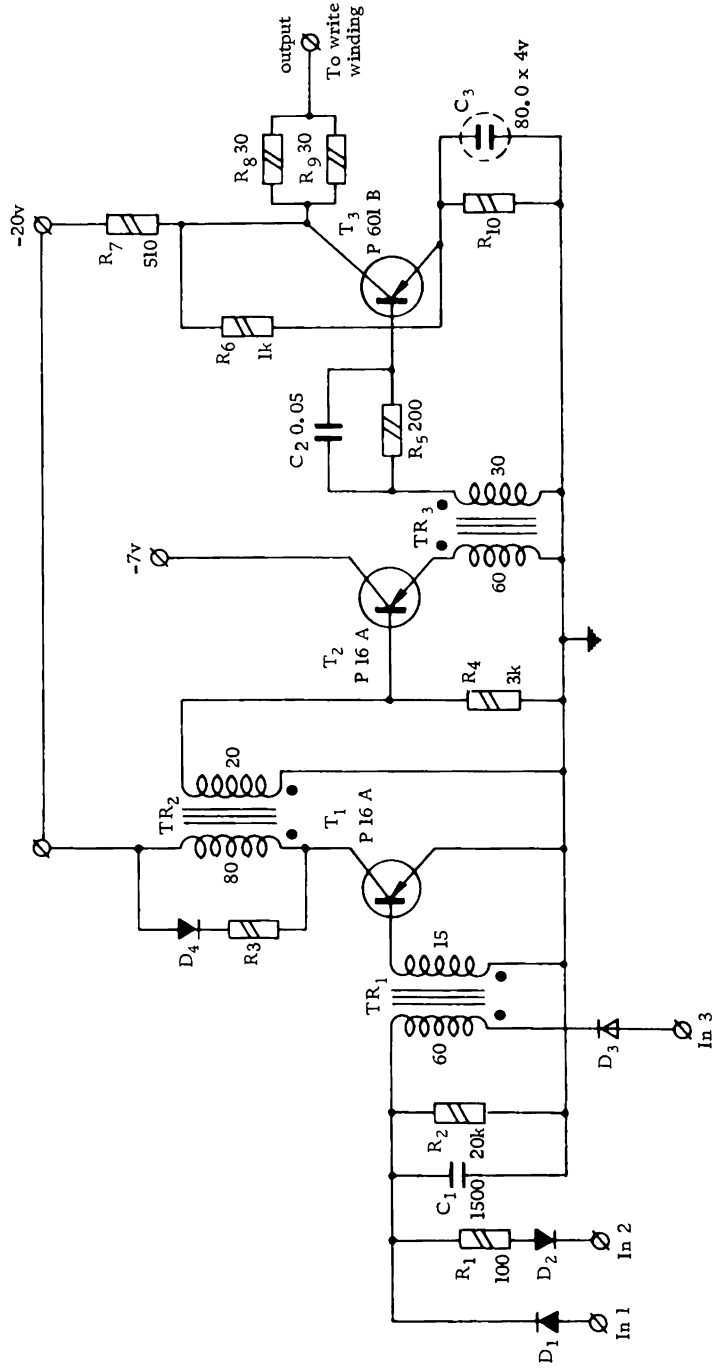


Fig. 172. Schematic of a write address driver.

The schematic of a write address driver is shown in Fig. 172. The base circuit of transistor T_3 includes an RC filter (R_5C_2) which serves to limit the duration of the write current in the address line. The divider consisting of R_6 and R_{10} gives at the emitter of T_3 a small negative voltage cutting off the transistor and eliminating the noise in the circuit of the write windings. Resistors R_8 and R_9 serve to limit the write current in the address lines to the required value.

8. DIODE-CAPACITOR REGISTERS

A register can be used to convert a serial code arriving from the integrator into a parallel code for subsequent operations in the memory. The magnetic internal memory of a DDA requires two registers. One register is a serial-to-parallel converter, and the other is an intermediate register for controlling the bit write amplifiers.

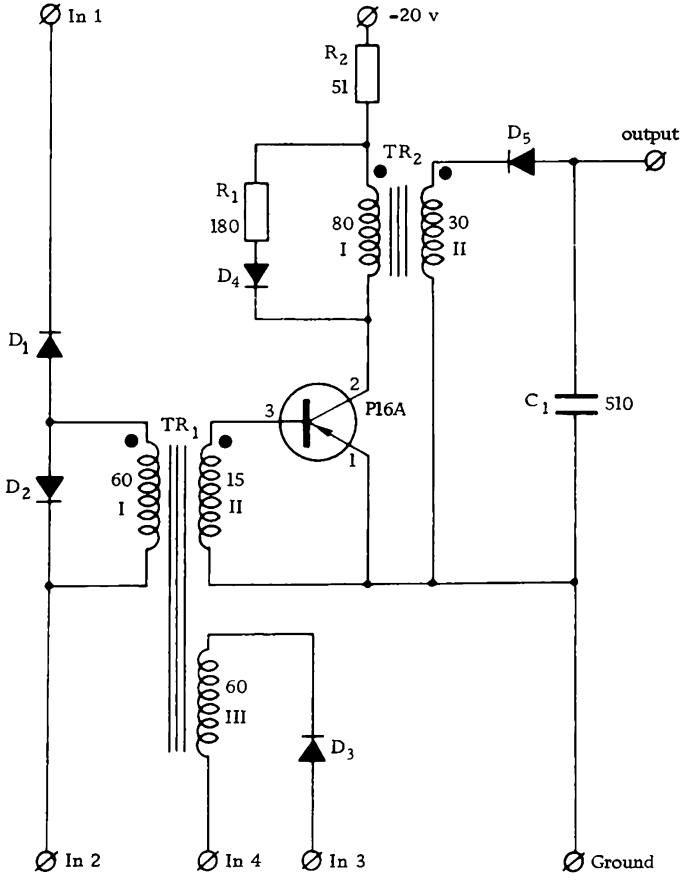


Fig. 173. One stage of first diode-capacitor register.

The first register is a diode-capacitor register. Figure 173 shows one stage using a diode-transformer circuit as a coincidence gate and a capacitor as the storage element.

A gated-pulse amplifier using a P16A transistor serves as a driver for charging of capacitor C_1.

Inputs 1 and 2 of the diode-transformer coincidence circuit are for the bit pulse from the DDA control unit and the pulse from the adder of the integrating unit.

Diode D_2 shunts the primary winding of transformer TR_1, removing spurious signals at the transistor base.

When transistor P16A is in saturation, the pulse at its collector has a duration of the order of 1-2 μsec. The pulse from the secondary winding of transformer TR_2 charges capacitor C_1 to approximately 6.5 v since there is a negative bias voltage of -6.5 v. Resistor R_2 in the collector circuit improves the pulse shape. Diode D_5 in the capacitor charging circuit must have $R_{rev.} \geqslant 1$ megohm. The potential of capacitor C_1 must be maintained at maximum for 20 μsec (i.e., for one operating period of one integrator of the DDA). Thereafter, a clear pulse (through the common power amplifier) transfers the number code to the second register. The clear pulse corresponds to a timing pulse. If, at the moment of application of the clear pulse, capacitor C_1 was charged (code 1 in the given bit position), this pulse is isolated at the transformer and the capacitor is simultaneously discharged. Thereafter, the first register is again prepared to receive the serial code arriving from the adder during operation of the next integrator.

The second register (Fig. 174) is similar in operating principle to the first. The only difference lies in the fact that transistor T_1 (type P16A) operates as a gated amplifier with a voltage-pulse coincidence circuit. Storage capacitor C_1 of all stages of this register is simultaneously charged, whereas in the first register the potential of the capacitor is of different duration depending on the time of the incoming pulses P_1.

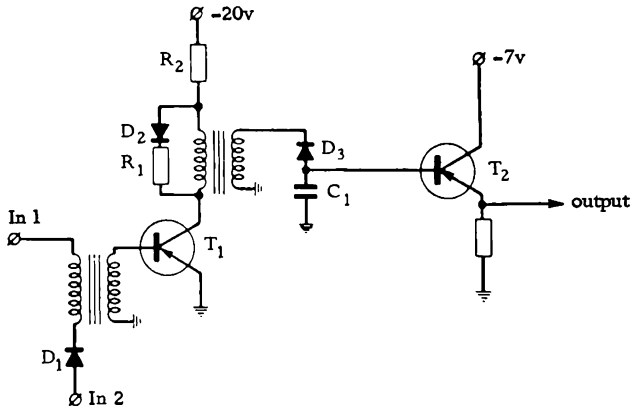


Fig. 174. One stage of second register.

The potential of storage capacitor C_1 of the second register is connected through the emitter follower of transistor T_2 (type P16A) to the input of the enabling and inhibit circuits of the write amplifier. The emitter follower gives optimum matching of the high output impedance of the second register stage with the low input impedance of the write amplifier circuit. The second register is cleared simultaneously by pulse P_{11} from the power amplifier for the entire register.

The power amplifiers delivering the pulses for clearing of the first and second registers consist of transistors of type P601 with emitter followers using P16 transistors for driving the P601 transistors. The emitter followers are resistance loaded and coupled with the P601 transistor through a capacitor which improves the stretching of the high-power clearing pulses.

9. WRITE AMPLIFIER

In a Z-type (coincident-current) memory with core switching matrix the bit write winding requires a current I_{bw} whose polarity depends on whether the bit to be written is given as code 1 or code 0. In order to obtain output pulses of different polarities from the write amplifier, the potential at the emitter follower of the second register stage of the memory is taken as the input to the inhibit circuit. Figure 175 shows the first stage of the bit write amplifier, which is constructed on the voltage-pulse principle.

Transistor T_1 (type P16A) is used as an emitter follower. The potential from the emitter follower is applied to the diode-transformer inhibit circuit, which has two pulse inputs. To these inputs are applied simultaneous pulses of different polarity: a negative pulse to input 0 and a positive pulse to input 1.

When the potential at resistor R_1 is low (of the order of 7 v), it corresponds to code 1 of the bit position in the second register. A positive pulse is applied to input 2 and passes through diode D_2. A negative pulse arises at the secondary winding of base transformer TR_1 and drives T_3 to saturation.

During this period transistor T_2 remains cut off. In the presence of code 0 in the given bit of the second storage register, a high potential (close to 0) occurs at the emitter of T_1. A negative pulse arises and is applied to input 1 of the inhibit circuit. Transistor T_2 is driven into saturation and T_3 is cut off.

The emitter follower has a small emitter resistance (300 ohms). The output stage of the write amplifier is a push-pull power amplifier with high-gain transistors.

Transformer TR_3 generates current to drive the bit write winding. For best operation of the memory, it is desirable to have write current pulses I_{bw} with peak value approximately equal to one-third of the base current. This eliminates the possibility of destruction of information, which may occur with repeated application of the bit write current to the storage elements due to creeping of the operating point on the hysteresis curve.

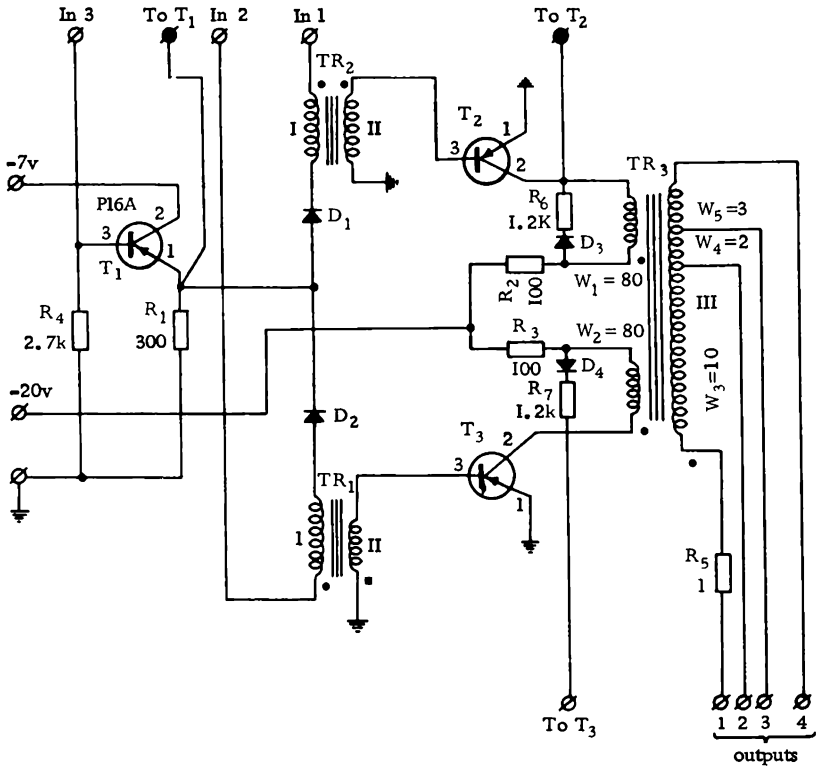


Fig. 175. Bit write amplifier.

Ferrite cores with dimensions $12 \times 5 \times 5$ mm. are used for pulse-shaping in the output transformer. Although the current pulses obtained from this transformer have a slight backswing, they do not impair the memory operation.

Pulses applied to the bases of transistors T_2 and T_3 (type P16B) cause them to conduct. These pulses have an amplitude of the order of 2 v and drive the transistors to deep saturation. The pulse duration at the collectors, connected to a common transformer load, is 4 μsec during saturation. The current pulse in the secondary winding has the same duration. For optimum shaping of the collector pulses as well as for limiting of the collector current in the load circuit of each transistor, there is a limiting resistor R of 100 ohms. An additional resistance of the order of 1 ohm is connected to the output circuit to decrease the time constant of the output discharge and to stabilize the discharge.

10. READ AMPLIFIER

A read amplifier is shown in Fig. 176. It amplifies and shapes the pulses induced in the read windings of the memory array

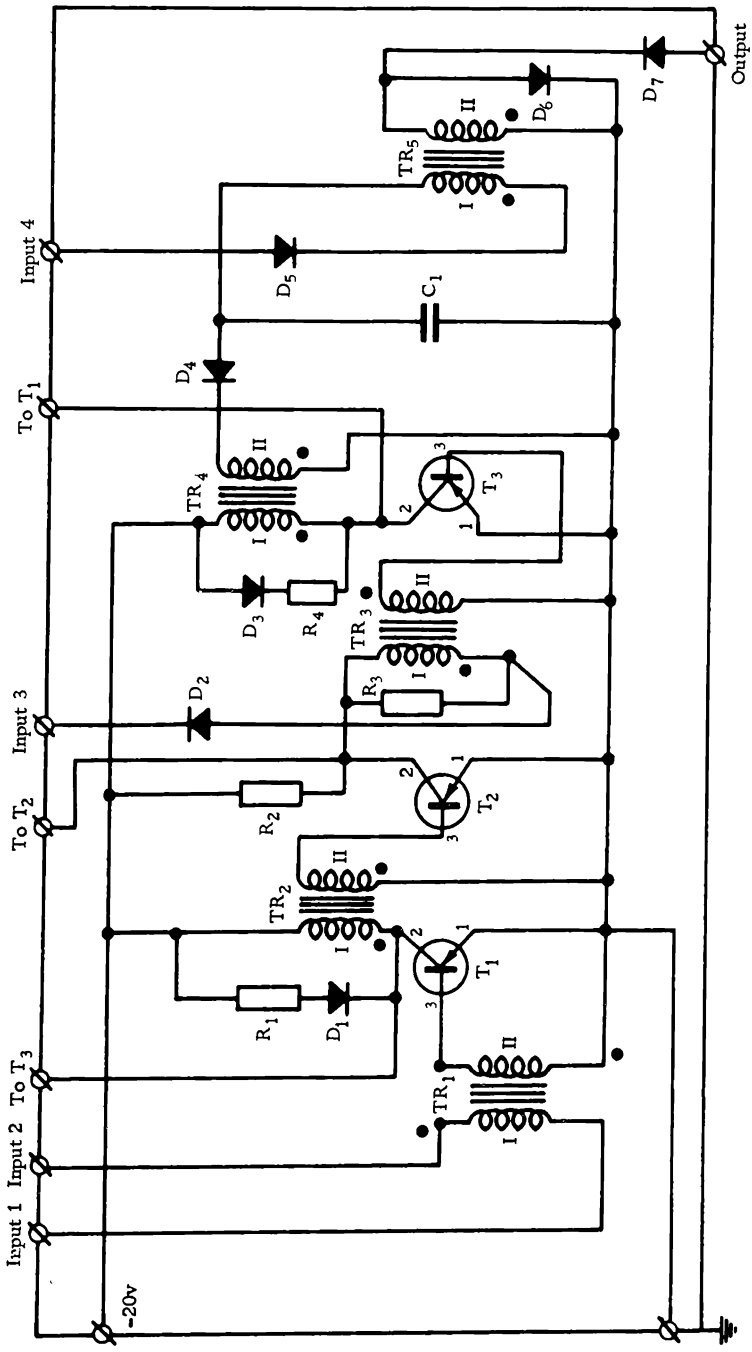


Fig. 176. A bit read amplifier.

during read-out. For ferrite cores of type VT-1, the emf induced in the read winding is of the order of 100-150 mv. In order to isolate the transients due to the write current, it is necessary to gate (or strobe) the amplified output pulses.

The first two stages of the read amplifier use transistors of type P16B and the third stage uses a transistor of type P16A. The first stage amplifies the negative output pulse from the read winding. Positive pulses are not amplified. The read winding is a-c coupled to the base of the first transistor by a step-up transformer (with ratio of 3). The second stage, a pulse stretcher, has a high collector resistance. The pulse duration at this stage is 3 to 4 μsec. These wide pulses have short leading edge and are shaped at the collector. The collector is connected to one of the inputs of a diode-transformer coincidence circuit. A read strobe pulse which is common to all read amplifiers is applied to the other input of this circuit. The strobe pulse has an amplitude of approximately 10 v and is shaped by a separate power amplifier. Since pulse P_{19} is applied to the power amplifier, the read strobe arrives 1 μsec after the read control signal is applied to the address circuits. This delay is provided for gating the amplified read-out pulse in the stretcher stage, since time is required for switching the selected cores of the array (about 1 μsec). The read strobe pulse passes through the coincidence circuit and, at the base of the third stage (a gated pulse amplifier), gates the read-out pulses. The pulses which result from core switching by the write currents and are then amplified by the first two stages are not gated through. The pulses at the collector of the gated-pulse amplifier have a duration of 1.5 μsec with a rise time of 0.1-0.2 μsec. The output winding of the gated-pulse amplifier is connected to a stage of the read register, which is also used to convert the parallel output of the memory into a serial input for the serial adder of the integrating unit.

Each stage of the register has a storage capacitor and a diode-transformer coincidence circuit connected to input 1 of the read amplifier. The storage capacitor is charged by the output pulse of the read amplifier to a potential of -7.5 v, which is determined by the bias applied to input 4 of the diode-transformer coincidence circuit. To this same input there is applied a positive bit pulse (7 v) from the bit counter of the control unit.

Bit pulses P_1—P_{18} are applied in time sequence to each register stage, beginning with the first and ending with the 18th. If code 1 is read out from a given bit position, the read amplifier charges the storage capacitor of the register stage and, upon application of the bit pulse corresponding to the given bit position, a pulse appears at the output of the coincidence circuit and reaches the output amplifier of the memory through diode D_7. At the same time a bit pulse discharges the storage capacitor of the stage to approximately zero potential. In read-out of code 0 the storage capacitor retains zero potential and the bit pulse does not pass through the coincidence circuit (i.e., the output signal is zero). Storage duration at the capacitor varies, depending on the time of

the incoming bit pulse; its maximum duration is 20 μsec. Reliable clamping of the capacitor voltage requires that D_5 have $R_{rev.} \leqslant 1$ megohm. Diode D_6 removes the transient at the output.

A read amplifier using p-n-p and n-p-n transistors with direct interstage coupling is shown in Fig. 177a. Direct interstage coupling is used in order the decrease the time constant of the amplifier. The input signal which comes from the read winding of the memory has an amplitude of 20-25 mv with negative polarity for code 1 and positive polarity for code 0. The amplifier consists of three d-c coupled transistor stages (T_1-T_3). The first stage (T_1) is a linear amplifier. A bias voltage of +12 v is applied to the emitter to compensate the negative voltage drop at resistor R_1. This negative voltage tends to cut off the transistor, but, at zero input voltage, maintains a small positive potential at the emitter to cause the transistor to conduct. Emitter resistor R_1, serving to establish the operating point, is shunted by capacitor C_1, which decreases the resistance in the emitter circuit for the input pulse. Cutoff of the input stage of the read amplifier during write operation is achieved by applying to the emitter of T_1 a negative blocking pulse through diode D_2 and resistor R_2. Under the influence of the blocking pulse, the potentials at the upper and lower plates of capacitor C_1 are approximately -3 v and -5 v, respectively. Upon completion of the blocking pulse, capacitor C is rapidly discharged through the emitter of T_1 and the base potential of T_1 is at this time equal to zero. The -1.5-v voltage source limits the lower voltage level of diode D_1. The -9-v source establishes an operating point at the transistor emitter for the current through resistors R_1, R_2, R_3 and also limits the lower voltage level of diode D_2.

The output voltage of the first stage is applied to the base of the second stage (an emitter follower). Let us assume that a negative pulse is applied to the input of T_1. The T_1 collector current increases and T_2 is driven further into saturation. The emitter potential increases to +2 v, with the result that diode D_4 is cut off. Capacitor C_2 stores the voltage existing at the emitter of T_2 until the arrival of the input signal. As long as no strobe pulse arrives at point m, T_3 and D_5 remain cut off. Upon arrival of the strobe pulse diode D_5 conducts and the emitter of T_3 is at the voltage of capacitor C_2. Since the emitter potential of T_2 is increased by +2 v, transistor T_3 conducts. A negative pulse appears at the output of T_3.

If a positive pulse (code 0) from the memory read winding arrives at the input of T_1, T_3 will be cut off when the strobe pulse passes, since the voltage at the emitter of T_2 becomes more negative after passage of the pulse.

Zero drift of the d-c amplifier is here eliminated because the input voltage at the base of T_3 is proportional to the difference between the voltage at the moment of read-out and the voltage stored at capacitor C_2 during the 1 μsec before read-out. The output signal of the read amplifier appears upon coincidence of the read signal in the memory winding and of the strobe pulse at point m. Diode D_5 at the amplifier output grounds noise pulses of

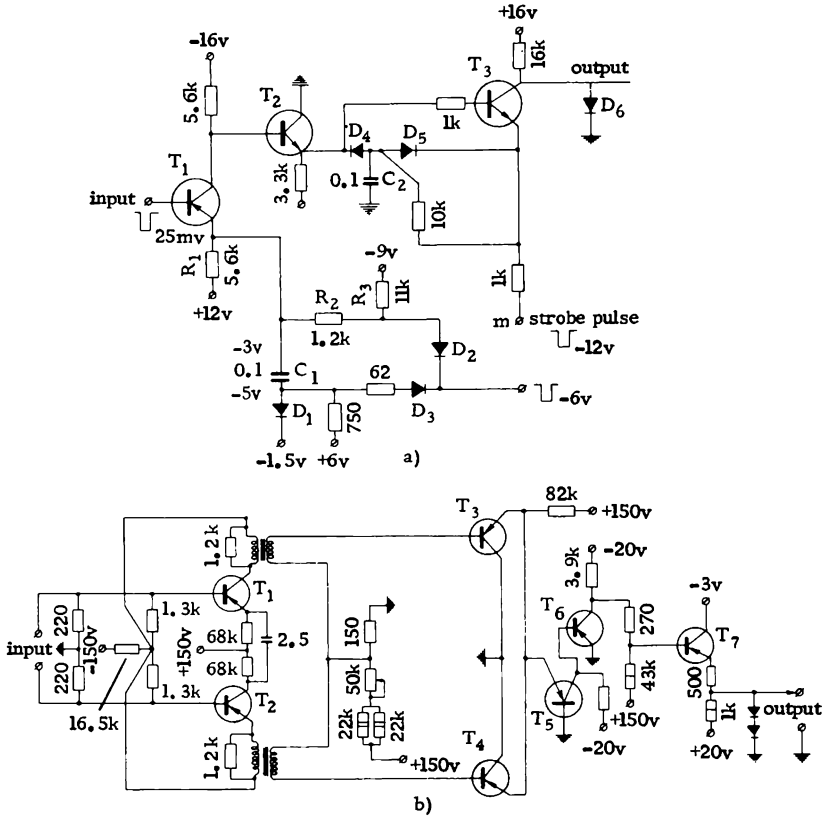


Fig. 177. Transistor read amplifier.

positive polarity. The amplifier acquires sensitivity upon disappearance of the blocking pulse at 2 μsec after termination of the write pulse. Since the write pulse in the memory is completed 9 μsec before the beginning of readout, the read amplifier blocking signal may either be completely absent or occur only during the write time.

The first stage (T_1 and T_2) of the read amplifier in Fig. 177b is a differential amplifier and is transformer-coupled to the second stage (T_3 and T_4). For reliable cutoff of these transistors, the bias voltage obtained from a divider connected to the +150-v source is applied to the secondary windings of the transformers. Transistors T_3—T_4 are connected in parallel as emitter followers with a common emitter resistor of 82 kilohms.

The transformer secondary windings in the collector circuits of transistors T_1 and T_2 are so connected that, in the presence of an input signal, at the bases of transistors T_3 and T_4 there arises a negative pulse simultaneously causing both transistors to conduct, with

the result that the signal strength is doubled. The signal from the emitter follower output is further amplified by transistors T_5, T_6 and T_7. In the absence of an input signal transistor T_5 is in the conducting state and almost all the voltage from the +150-v source is applied to the 82-kilohm emitter resistor. When a signal is applied to the amplifier input, transistors T_3 and T_4 conduct while the emitter potential of T_5 becomes close to zero. Hence transistor T_5 is cut off and transistors T_6 and T_7 are switched on. The collector voltage of transistor T_6 changes from -4 v to 0 when the voltage corresponding to code 1 is applied to the amplifier input, and the output of the emitter follower T_7 changes from -3 to +1 v for triggering the flipflop. The resistor divider in the emitter circuit of T_7 provides the required voltage bias and the diodes connected in parallel in the output clamp the positive output voltage.

11. DIODE-CAPACITOR READ REGISTER

As a read register for storing a read out word we may use diode-capacitor storage elements in which the code is stored in the form of an electrical charge at a capacitor with a capacitance of several thousandths of a microfarad.

Each stage of the register element (Fig. 178a) consists of a capacitor and a symmetrical diode gate. Diodes D_1 and D_2 are normally cut off and capacitor C is discharged. During the write operation symmetrical pulses of opposite polarity from the secondary windings of transformer TR switch diodes D_1 and D_2 to the conducting state and a write pulse U_w is applied to the lower plate of capacitor C for entry of the number code. At this moment the potential of the upper plate of the capacitor is equal to zero while the potential of the lower plate is equal to $+U_w$. The U_w pulse does not terminate until after pulses U_a and U_b. On termination of pulses U_a and U_b the upper plate of the capacitor, now isolated from ground, is still at zero potential and the lower plate at potential $+U_w$. After termination of the U_w pulse the potential of the lower plate becomes equal to zero and the potential of the upper plate reaches $-U_w$; thereafter the capacitor is slowly discharged through the leakage resistance of back-biased diodes D_1 and D_2.

The stored information is represented by the presence (code 1) or absence (code 0) of charge at the capacitor. To read out the stored information, pulses U_a and U_b are applied again. The capacitor discharge current through resistor R produces a voltage pulse as the output.

If the information must be stored for more than several tenths of a microsecond, the stored charge at the capacitor must be regenerated.

The above described register is designed for temporary storage of a single 16-bit number for a period of up to 40 μsec. Parallel-to-serial word conversion is provided. Serial transfer of a number from the register is achieved at a frequency of 1 Mc. Figure 178b shows the schematic of a 16-bit register using diode-capacitor elements.

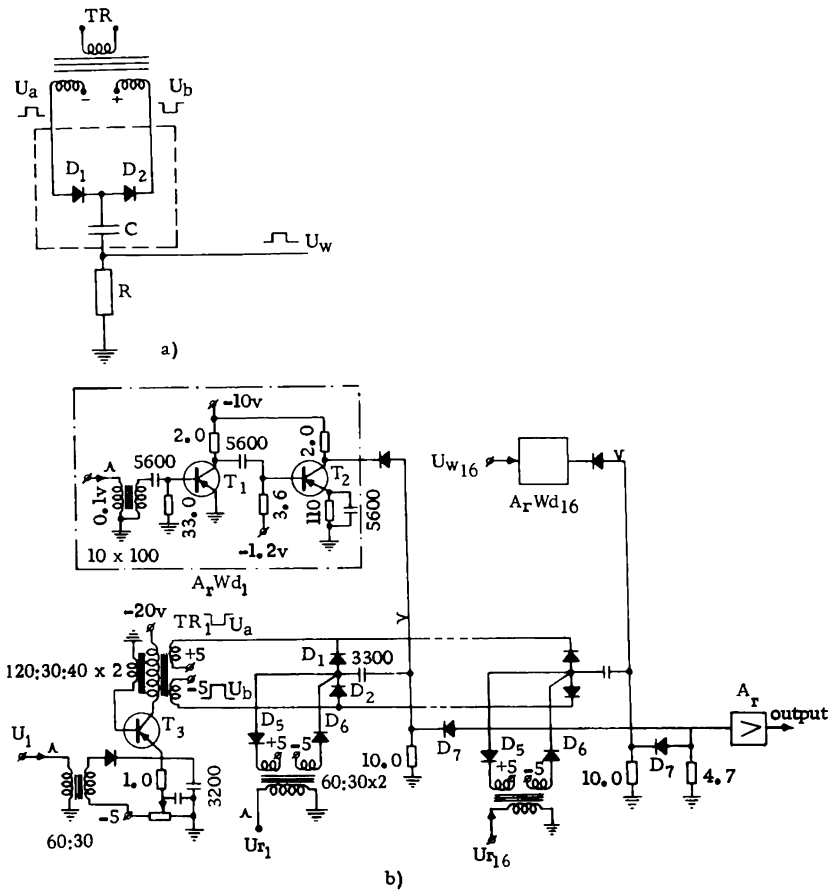


Fig. 178. Diode-capacitor storage element and register: a, element; b, register.

Each bit is stored in a storage element consisting of four diodes D_1, D_2, D_3 and D_4 of type D2-D and a 3300-$\mu\mu$ fd capacitor. During the write operation pulses U_a and U_b, which unblock gates D_1 and D_2 for all 16 bit positions, are generated by a blocking oscillator (transistor T_3). Pulse U_1 triggers the blocking oscillator. The word is transferred from the memory proper to the register through 16 read amplifiers (A_rWd). Amplitude of the pulses at the A_r input is 0.05 v. At the A_rWd output we obtain after amplification a pulse of the order of several volts. The timing and duration of pulses U_a and U_b and the write pulse U_w are so chosen that the U_w pulse terminates upon conclusion of pulses U_a and U_b. For code 1, a positive charge is stored at the capacitor.

During read-out from the register, gates D_1 and D_2 of all bit positions remain cut off while gates D_5 and D_6 are opened in sequence at intervals of 1 μsec under the action of the U_w pulses.

At a bit position where gates D_5—D_6 are unblocked, the capacitor at that position is charged through a 10-kilohm resistor at the bit-position bus. The pulses of all bit positions are applied in parallel to a 4.7-kilohm resistor; thereafter, they are amplified by the write amplifier. The pulses read out from the register have a duration of 0.3 μsec and amplitude slightly larger than 1 v.

12. MEMORY NOT USING CURRENT-COINCIDENCE PRINCIPLE

Present ferrite-core memories employ the principle of coincidence of half-select currents, whereby word selection is achieved for the write or read operations. This principle has the disadvantage that stringent requirements are imposed on the amplitude of half-select current and on the quality of ferrite cores (the squareness ratio of hysteresis loop).

For a small memory with a capacity of about 50-100 words this disadvantage is eliminated by the use of diode-transformer coincidence circuits. This results in a considerable increase in the operating reliability of the memory and in relaxing the requirements imposed on ferrite core quality. In the memory of Fig. 179, clearing or setting of the selected cores to the 0 state during read and write operations is achieved in the same manner as in the above described memory. Clearing is achieved by applying a pulse to all address lines of the 64 integrators, while for subsequent write operations pulses from the 16 write amplifiers are applied to the 16 vertical lines. To write code 1 in a given bit position of a number, a positive pulse is applied by the corresponding write amplifier. To write code 0, no pulse is applied by the write amplifier.

The use of a separate transformer driving each ferrite core avoids the necessity for coincidence of half-select currents in the ferrite core; instead, an external coincidence circuit is provided in the diode-transformer matrix.

Each write amplifier draws an insignificant power for switching of only one ferrite core. However, the amplifiers feeding the 64 address lines (integrators) must deliver sufficient power for simultaneous switching of one row of 16 ferrite cores.

The transformers permit reducing the voltage in the secondary winding to 1 v and increasing the switching current by several times. This also permits extremely efficient use of the power obtained from a transistor power amplifier.

Read-out of the information stored in the memory is achieved in the same manner as in the above described arrangement.

A disadvantage of the proposed writing method is the need for one transformer and one diode for each ferrite core. In order to write 64 16-bit words, 1024 transformers and 1024 diodes are required. Since it is necessary to write two words simultaneously in a digital integrating machine, two core arrays (each

having 1024 diodes) are required. For this reason, the above described writing method is economical only for a small number of words (integrators).

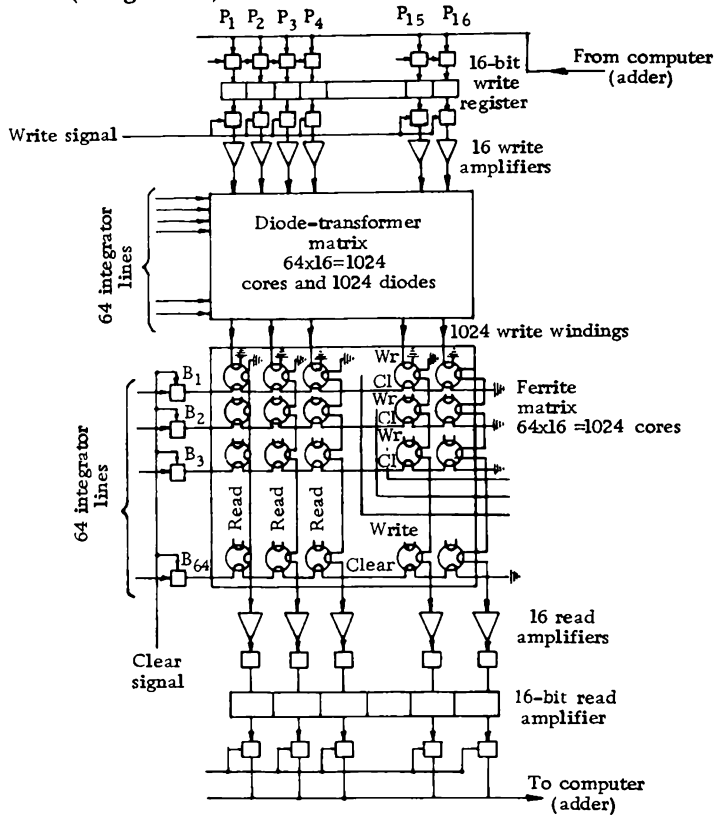


Fig. 179. MIM not requiring half-select current coincidence.

13. MAGNETOSTRICTIVE DELAY-LINE STORAGE

Magnetostrictive delay lines are simple, inexpensive and durable devices possessing a low temperature coefficient. They may be used as an internal memory for a DDA.

The magnetostrictive delay line shown in Fig. 180 is a nickel wire (or rod of several fine nickel wires) on which are placed two small coils. Due to the magnetostrictive effect, a current pulse applied to the drive winding creates two ultrasonic waves of mechanical stress (forward and reverse) propagating in opposite directions. One of these waves (reverse) is damped by a soft rubber or neoprene termination; the forward wave is propagated along the wire with the velocity of sound in nickel. This wave is received at the other end of the line by a receiving coil. The line

section under the read coil is magnetized by a permanent magnet *NS*. Upon arrival of the wave of mechanical deformation at the receiving (read) coil, an inverse magnetostrictive effect causes a change in magnetic flux in the nickel within the receiving coil. An emf proportional to the magnitude of mechanical deformation is induced in this coil. Since the mechanical deformation is slight, the emf is also small and amounts to a few tens of millivolts (in the best case, a hundred mv).

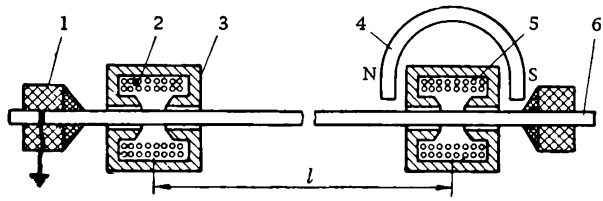


Fig. 180. Magnetostrictive delay line: 1, soft rubber or neoprene termination; 2, drive windings; 3, ferrite core; 4, magnet; 5, read winding; 6, nickel line.

The magnetostrictive effect in nickel consists in a small change in length of the wire (contraction) under the influence of a magnetic field. Contraction of the wire causes a longitudinal wave of mechanical stress propagating along the wire with the velocity,

$$v = \sqrt{\frac{E}{\varrho}},$$

where E is Young's modulus (for nickel, $E = 21.4 \times 10^{11}$ dyne/cm^2); ϱ is the specific density (for nickel, $\varrho = 8.9$ g/cm^3); v is the velocity of sound (in nickel, $v = 4.91 \times 10^5$cm/sec or 2.08 μsec per cm of delay line).

The delay time t is,

$$t = \frac{l}{v} = 2.07\, l\ \mu\text{sec}$$

where l is the length of the wire in cm.

The mechanical stress arising in the nickel under the influence of the current pulse in the write or exciting coil (caused by the change in magnetic flux) is approximately proportional to the first time-derivative of the input voltage, since the change in magnetic flux is proportional to the rate of current change in the write coil. The emf arising in the read coil, in turn, is proportional to the change in magnetic flux in this coil. The change in magnetic flux of the read coil is proportional to the change in mechanical stress, or proportional to the first derivative of the wave of mechanical stress. Thus, the emf in the write coil is proportional to the second derivative of the input signal.

For the most complete conversion of magnetic energy into mechanical energy and vice versa, both coils (the exciting and read) must lie as close as possible to the nickel wire and measures must be taken to decrease the leakage flux. For this purpose, the coils in Figure 180 are surrounded by a magnetic conductor and the coils themselves are of small dimensions (wire diameter 0.03 mm with 200-700 turns). The principal attenuation of the input signal during transmission through the magnetostrictive delay line occurs in the conversion of magnetic energy to mechanical energy and vice versa. Due to the low coefficient of electromechanical coupling of the excitation and read coils, the attenuation in transmission of the signal is 35-60 db. Attenuation of the ultrasonic wave in the nickel wire is relatively small. Small distortions arise in propagation of longitudinal waves of stress along the nickel wire due to the increase in damping with an increase in frequency. In addition, distortions occur due to the necessity for bending the wire in actual construction (in the form of a spiral) in order to decrease the size of the memory. These distortions limit the maximum delay time for a magnetostrictive line, or the storage capacity for a given pulse frequency.

Special measures must be adopted to decrease the influence of the reverse wave and the echo signal occurring upon its reflection from the support points. For this purpose the ends of the line are clamped between two soft rubber or caoutchouc holders to damp the oscillations. In addition, the line may be made of several thin nickel wires (up to 100 with diameter of 0.05-0.08 mm) twisted together in the form of a single rod or many thin nickel ribbons threaded through the two coils. As a result of the effect of incoherent reflection in such a bundle of ribbons or wires line attenuation is decreased and distortion due to reflection of the waves is eliminated. Moreover, emf in the read coil is increased due to the larger diameter of the rod. Because of distortion in a magnetostrictive line and since the bandwidth is only one half that of the mercury line, pulse repetition frequencies are limited to 300-500 kc. At these frequencies, the time interval between two pulses is 2-3 μsec, (i.e., one centimeter of nickel wire may store one bit). In order to store a binary number having, for example, 72 bits, a nickel rod 72 cm long is required. With a long nickel wire, there is a sharp increase in signal distortion due to the occurrence of reflected waves, echo signals, etc. Hence magnetostrictive lines are used in the form of short segments of nickel wire not longer than a few meters and having capacity of no more than 100-200 bits. As an example, let us present the data for one of the lines (see also Table 9):

Delay time, μsec	90
Line attenuation (insertion loss), db	35
Bandwidth, kc	300
Midfrequency of passband, kc	200
Nickel ribbon line, 99% Ni, dimensions in mm	0.2 × 0.5

Coils:
(a) Transmitting, 200 turns, coil length 4.8 mm
(b) Receiving, 100 turns, coil length 3.2 mm

Resistance, ohms:
input 500
output 300-500

Current in shorted read winding with output resistance of 300 ohms, in ma. . . . . ~20

TABLE 9

Characteristics of a Nickel Line in the Form of a Thin Tube

Designation	Characteristics
Material	Nickel tube with outside diameter 0.125 inch and wall thickness 0.004 inch with slot to decrease Foucault currents
Delay per 2.5 cm of length	5.27 μsec
Input pulse duration	1.3 μsec (optimum 2.5 μsec)
Output pulse duration	Not less than 2.5 μsec
Transmission losses (signal attenuation)	40-50 db
Attenuation in nickel line	6 db per 100 μsec
Change in delay with temperature	0.16% per 1°C
Repetition frequency	350 kc (for a tube with diameter of 0.045 inch and wall thickness 0.002 inch, up to 600 kc)
Signal-to-noise ratio	10:1
Current pulse in drive winding	0.15 amp (35 ampere-turns)
Length of nickel tube	100 cm
Interval between pulses	1.67 μsec for 100-μsec delay line

The temperature dependence for nickel lines is small. The delay time t of the line as a function of temperature T may be approximately expressed by the formula

$$\frac{\partial t}{\partial T} = \frac{l}{2vE} \cdot \frac{\partial E}{\partial T},$$

where l is the length of the line; E is Young's modulus; v is the velocity of propagation. For nickel the quantity $\frac{l}{E} \cdot \frac{\partial E}{\partial T} = 10^{-3}$ per 1°C.

For a nickel line with delay $t = 100$ μsec and with a temperature change of 20°C, the change in delay time is 0.5 μsec. This amounts to 0.25 μsec per cm of delay line per 1°C.

Figure 181 is a diagram showing electronic components associated with a magnetostrictive delay line. A write amplifier with pulse power of the order of 10 w drives the write coil. A pulse arrives at the write amplifier input from the write flipflop, which is set to the 1 or 0 state by a code pulse from an external input or from the feedback loop. The pulse arriving externally is gated by a CP pulse at gate B_1. The amplifier for reading weak signals from the read coil is chosen with a gain of 250 to 7000. The amplifier pulses set the read flipflop to the 1 or 0 state. The output of the read flipflop controls gate B_2 to pass the CP pulse to the feedback loop. Gate B_3 serves to clear (erase) the codes stored in the line. The vacuum-tube versions of the read and write amplifiers are shown in Figs. 182 and 183.

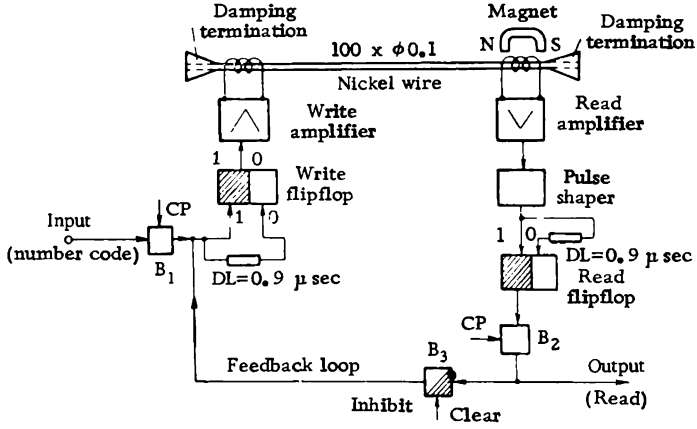


Fig. 181. Diagram of magnetostrictive delay line control.

Semiconductor components may be used for the read and write functions associated with a magnetostrictive line by the use of a high-power high-frequency transistor of type P601 (5 w, 2 Mc). The read and write amplifiers may employ the circuits previously

described in this chapter. Instead of read and write flipflops, storage of pulses may be achieved by means of a capacitor or a dynamic flipflop with a capacitor.

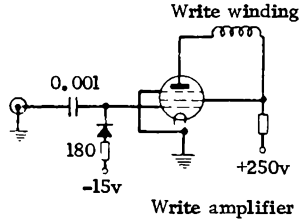


Fig. 182. Write amplifier for magnetostrictive delay line.

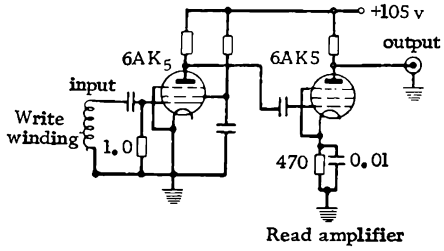


Fig. 183. Read amplifier for magnetostrictive delay line.

Compensation of the temperature dependence of the magnetostrictive line may be achieved by the use of a circuit similar to a temperature-control servo system in which the reference frequency of *CP* pulses is varied according to the change in delay time of the line. The simplicity of the temperature control circuit lies in the fact that the pulses arriving from the read winding are so gated that the position of the leading edge of the pulse is fixed. In this case the change in delay time of the line with temperature leads to a change in duration of the output pulses; the permissible variation in length of the line is of the order of 1 μsec.

In order to eliminate this temperature dependence, instead of nickel, we may use special alloys with extremely low temperature coefficient (for example, a filament of elinvar with nickel plating at the end). Permalloy alloys containing 65% Ni possess considerably better magnetostrictive properties than nickel, but they have not been used in magnetostrictive delay lines. In using magnetostrictive lines in a DDA, they must be connected in parallel (Fig. 184) in order to obtain adequate storage capacity. Each of the lines in this case is a section of wire approximately 1 m long with a storage capacity of the order of five 20-bit words (200 μsec). Readout of the information contained in each line is

achieved through gates B_1—B_n, each of which is unblocked in time sequence by integrator number lines. The write gates B'_1—B'_n, controlled by the same integrator number lines, are unblocked at the same time.

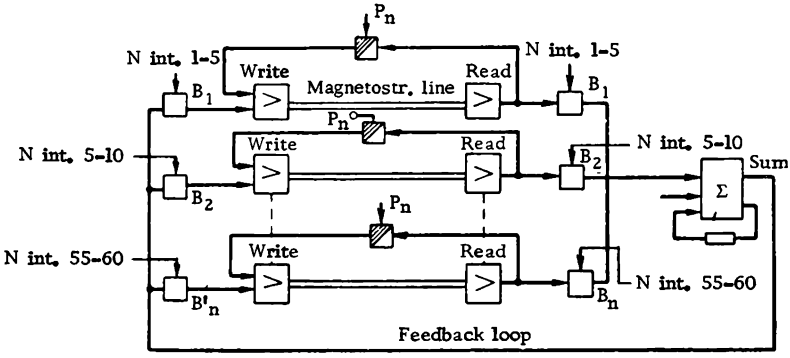


Fig. 184. Parallel connection of magnetostrictive lines in DDA.

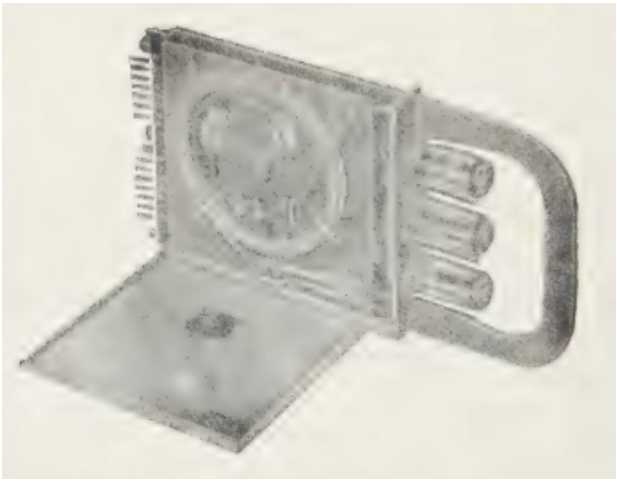


Fig. 185. A magnetostrictive line package.

Magnetostrictive delay lines (Fig. 185) may be used as standard storage registers in a parallel integrating computer. Each of the integrators requires two or three delay lines with a capacity of 40-60 μsec for a single binary word of 20-28 bits. With a pulse repetition frequency of 500 kc the time for one iteration is 40-60

μsec (i.e., from 1660 to 2500 iterations per second), depending on the number of bits. This rate is adequate for analog and control computers.

14. ACOUSTIC DELAY-LINE STORAGE

The need for an economical high-speed internal storage has led to the use of acoustic delay lines for storage of number codes. Acoustic delay lines (with solid or liquid medium) are made in the form of a tube or plate with two quartz piezo-converters at the ends (Fig. 186). At the input the quartz plate converts the electrical signal to an acoustic signal. A voltage pulse arriving at the capacitor plates between which the quartz plate is fastened causes the latter to contract and expand in proportion to the applied voltage. The mechanical signal is transmitted to the line and propagated therein with the speed of sound as determined by the density and elastic properties of the medium. At the output of the line, by means of quartz plate 2, the mechanical signal is reconverted to an electrical signal.

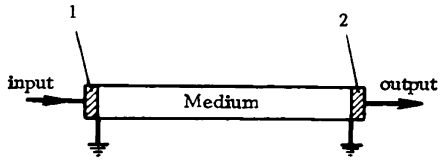


Fig. 186. Ultrasonic line.

The speeds of sound for several solid and liquid media which may be used for delay lines are listed below.

Medium	Ultrasonic velocity (10^{-5} cm/sec)
Magnesium	2.33
Mercury	1.46
Quartz	5.75
Water	1.43
Alcohol	1.44

Consequently, with a relatively short line we may obtain a time delay of hundreds and thousands of microseconds.

A modulation technique is usually employed to transmit a pulse with duration of the order of 0.5 μsec (repetition frequency of 1 Mc) through the delay line. The input pulse from the computer (Fig. 187a) is modulated by a carrier frequency (Fig. 187b) which is usually greater than the pulse repetition frequency by a factor of 10 (10 Mc in our case). The pulse-modulated signal is then

applied to the quartz plate. At the output of the line the pulse is detected, amplified, and regenerated in shape and phase.

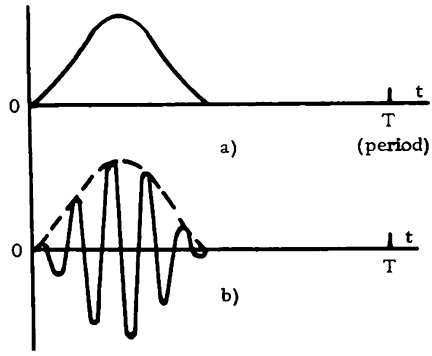


Fig. 187. Pulse modulation of signals.

The 1-Mc signal-pulse frequency and 10-Mc carrier frequency are not the maximum frequencies. For a fused quartz line, the input frequency may reach several megacycles. However, technical difficulties are encountered in operating at frequencies above 1 Mc.

The output pulse of the line, restored in shape and phase, is fed back to the quartz driver so that the pulse train circulates in the line for an unlimited period of time once stored in the line. This process is referred to as recirculation.

After being introduced into the delay line, the pulse will appear at the output a definite time later; this time is known as the delay time. The capacity of the line (i.e., the number of pulses which can be stored in the line) depends on the physical length of the line, the pulse frequency, and the speed of sound in the medium. For mercury, for example, the capacity of a line 1 m long is 685 pulses at a repetition frequency of 1 Mc.

The code pulses circulating in a line may be removed or transferred only when they appear at the output. The use of such a delay line as an internal memory requires a certain amount of waiting time to extract the word from it. The memory access time depends on the parameters of the line and on the moment of access to the memory. For example, with a repetition frequency of 1 Mc and a delay time of 360 μsec (eight 45-bit words) an average access time of 180 μsec is required for each reference to the memory. The relatively long access time is an important disadvantage of this type of memory. The access time can be decreased by decreasing the capacity of the line or by increasing the repetition frequency.

In today's computers using acoustic delay lines the clock frequency is usually 1 Mc, the carrier frequency 10 Mc, and the capacity of a single line 360-1024 bits. The number of lines in an internal memory may range from 16 to 64.

The temperature dependence of acoustic velocity in the line necessitates temperature control provisions. A servo control is

employed wherein the generator frequency follows the delay time in a special acoustic control line (or in a storage line with a location reserved for entering the control pulse).

Figure 188 shows the temperature control of an acoustic delay line. A counter divides the frequency of the pulse generator G by a number N (where N is the capacity of the line). The output pulse of the counter arrives at a quartz driver QD via coincidence circuit B. After passing through delay line DL, the pulse is amplified by amplifier A and arrives at one input of an AND coincidence circuit. The unbalance signal obtained at the output of the coincidence circuit is rectified by diode D and applied to the grid of reactance tube RT.

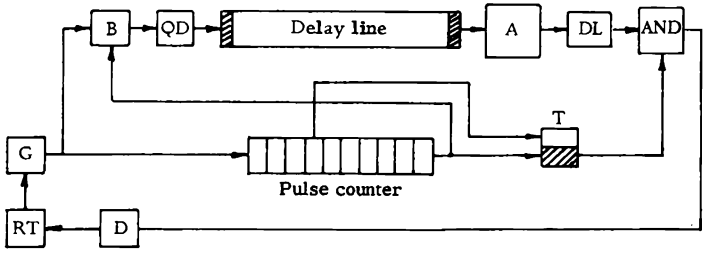


Fig. 188. Temperature control of an acoustic delay line.

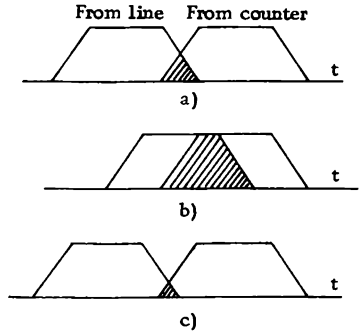


Fig. 189. Time diagram of pulses.

The diode-controlled reactance tube is connected in parallel with the oscillator of generator G. Capacitance of the tube depends on the voltage at its grid. The generator frequency must be so established that the trailing edge of the pulse of the delay line coincides with the leading edge of the pulse from flipflop T. The flipflop delivers a relatively wide pulse (of the order of several tens of microseconds). The counter is set to the zero state each time it counts 32 pulses. The AND coincidence circuit clamps the moment of coincidence of a pulse from the counter and a pulse from the delay line. If the delay time of the line is precisely equal to the nominal delay time, both pulses coincide as shown in Fig.

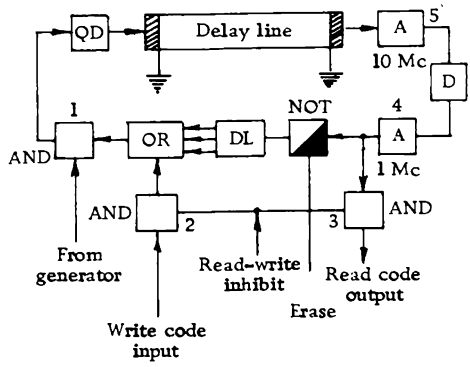


Fig. 190. Block diagram of code storage in ultrasonic delay line.

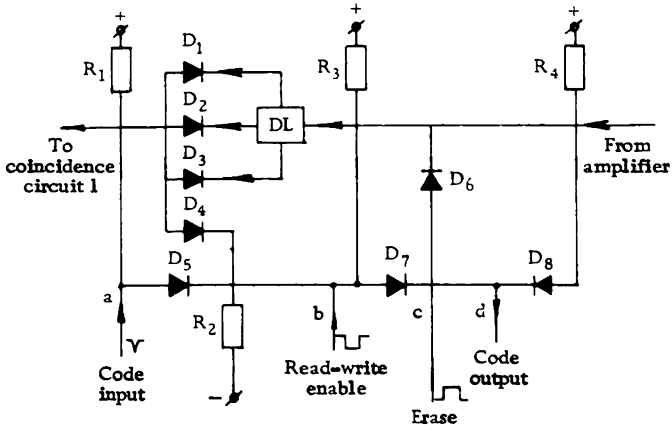


Fig. 191. Schematic of read, write and erase circuit.

189a and the coincidence circuit delivers at the diode output pulses whose amplitude is proportional to the shaded area. The resulting voltage at the diode is taken as the initial (zero) value.

Let us now assume that, due to a temperature change, the delay time of the line is increased. A pulse in the line will arrive at the coincidence circuit somewhat later than that from the counter (Fig. 189b). The amplitude of the output pulse of the coincidence circuit and of the diode increases and a larger voltage is applied to the grid of reactance tube RT. As a result, the capacitance of the tube increases, leading to a decrease in the frequency of the generator G. The generator frequency will decrease to such an extent that the signal at the output of diode

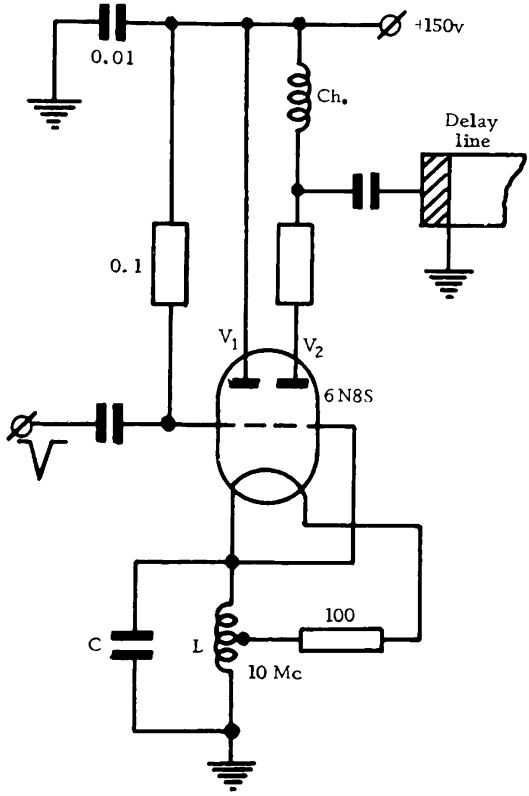


Fig. 192. Quartz driver QD.

D becomes equal to the initial value and the disturbance of balance disappears. An increase in delay time of the line is compensated by a corresponding decrease in generator frequency. A decrease in delay time of the line due to temperature (Fig. 189c) will cause a decrease in voltage at the output of the coincidence circuit and of the diode, which leads to a decrease in capacitance of the reactance tube and to an increase in the generator frequency. In short, the generator frequency is automatically controlled with the result that the same number of pulses can always be stored in the line.

For some acoustic memories another method of temperature control is used. The reference pulse generator GP is stabilized by a quartz crystal and the delay lines are placed in a temperature-controlled chamber.

Figure 190 shows the control elements for an acoustic delay line memory for erasing, writing (entry) and reading (output) of codes. The output pulse of the delay line is amplified, detected, shaped and applied through a logical NOT circuit to an electromagnetic line DL having a delay time of 1 μsec. In passing through

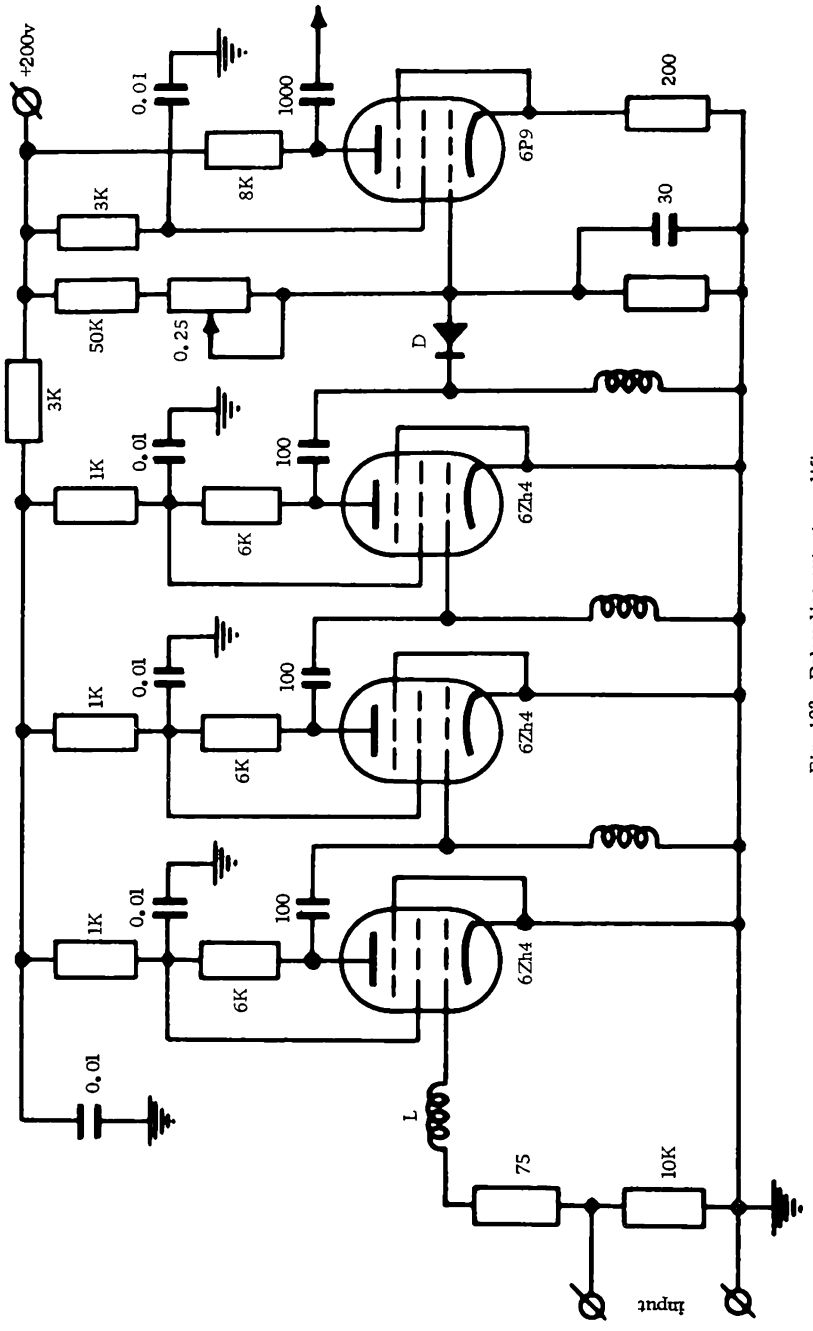


Fig. 193. Delay-line output amplifier.

line DL and the subsequent divider (OR circuit), the pulse from the acoustic delay line is "stretched" to 1 μsec. This wide pulse serves as a signal to the line through coincidence circuit 1 gated by the pulse generator. In this manner, a pulse from the output of the acoustic line is recirculated.

Stretching of the pulse increases the reliability of recirculation.

The schematic for read, write and erase circuits is shown in Fig. 191. From the output of amplifier 4, the signal arrives at delay line DL at a frequency of 1 Mc. The pulses from the three outputs of this line arrive in sequence at buffer diodes D_1, D_2 and D_3, resulting in stretching of the pulse applied to DL (through the OR gate).

Resistors R_1 and R_2 and diode D_5 form coincidence circuit 2 in Figure 190. Upon coincidence of negative pulses at the anode and cathode of diode D_5, a negative pulse corresponding to code 1 at point a is applied to the acoustic delay line. Diodes D_7 and D_8 and resistors R_3 and R_4 form a coincidence circuit 3. In the absence of a negative pulse at point b, diode D_7 conducts and diode D_8 is cut off. If a negative pulse is applied to point b, diode D_7 is cut off and diode D_8 conducts. The output signal at point d appears only when diode D_8 is cut off by a negative pulse from the read amplifier 4 (Fig. 190), i.e., upon coincidence of pulses from point b and from the read amplifier. The erase pulse passing through diode D_6 inhibits passage of the negative pulse from the read amplifier to the input of delay line DL.

The quartz driver QD (Fig. 192) is a Hartley sine-wave generator operating at a frequency of 10 Mc. It oscillates when a negative timing pulse is applied to the grid of triode V_1 at a frequency of 1 Mc. Amplification of the signal (several millivolts) at the delay line output is achieved by means of the amplifier shown in Fig. 193. The voltage gain of this amplifier is usually 10,000-20,000.

Chapter V

A Serial Binary Digital Integrating Computer

1. BLOCK DIAGRAM

The serial digital integrating computer shown in the block diagram of Fig. 194 is designed for real-time automatic control applications. This computer must have a memory of relatively large capacity for storage of various data. It must execute computations at high speed and possess high reliability, simplicity and economy of operation. In distinction from a digital computer designed only for computing purposes, the control machine must have numerous branching connections with a variety of input data sensors, measuring devices and control mechanisms. Hence for receiving inputs and for converting various input and output data, the machine must have input and conversion devices. The input data of the machine may be of three types:

a) Signals in the form of digital code from digital devices (counters, registers, etc);

b) Signals in the form of electrical voltages from various measuring devices or sensors (temperature, pressure, linear or angular displacements, etc). Such signals must be converted from analog to digital form by means of a multi-channel electronic converter.

c) Signals in the form of code pulses for connection, disconnection or switching of various mechanisms.

Insertion of digital coded data into the computer is achieved serially via each channel by means of gates controlled by bit pulses $P_1, P_2, P_3, \ldots P_n$. These gates are common to all digital channels. Sequential switching of the digital channels is achieved by gates $B_1 - B_{18}$, which are in turn controlled by those integrators specified at the beginning of each cycle of iteration for insertion of the input data.

The input channels for continuous signals are switched in the Input and Conversion Unit (see Chapter VII, Section 5). Insertion of various tabular or experimental functions is achieved by means of a Function Input Unit which may be built entirely of ferrite and semiconductor elements without the use of punched tape or magnetic tape. Both the Input and Conversion Unit as well as the

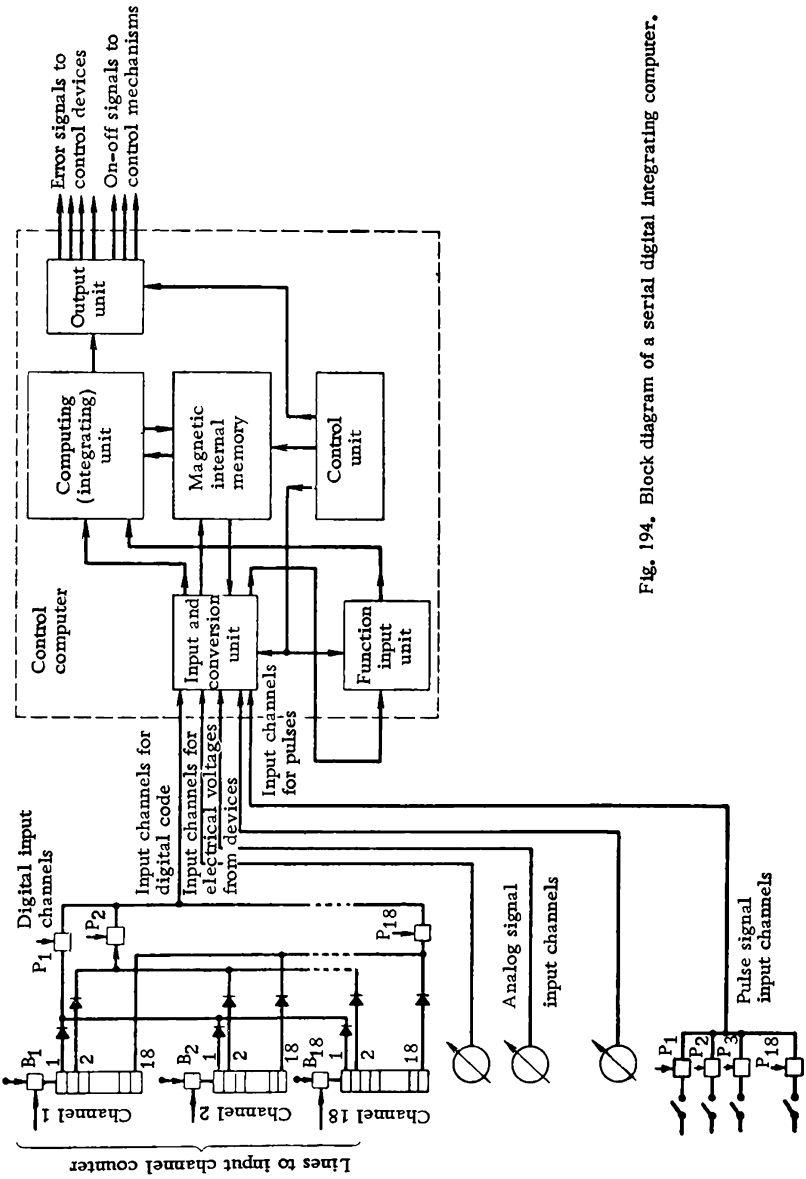


Fig. 194. Block diagram of a serial digital integrating computer.

Function Input Unit are integral parts of the computer since they use the control, storage and integrating units of the computer.

The computing (integrating) unit performs integration operations and certain logic operations according to the given program. It consists chiefly of two or three single-bit adders (depending on the method of integration); a $\Sigma \Delta y$ increment counter; a high-speed, small-capacity memory for storage of the integrator overflow pulses and increment pulses from the external input channels (in the form of registers or delay lines); and a number of gates and flipflops for execution and control of the integration operations. The data from the Input and Conversion Unit or from the Function Input Unit (in the form of increments) are inserted directly into the integrating unit (without first being stored in the magnetic internal memory) and are stored in high-speed registers or lines. This results in a considerable increase in the data input rate. The pulse repetition frequency is about 1-2 Mc. Insertion of external data in the form of increments considerably decreases the space required for the high-speed internal storage and increases the total amount of information which can be inserted into the machine. The total capacity of the internal storage also includes the data in the Function Input Unit and the data arriving at the Input and Conversion Unit.

The magnetic-core internal memory (MIM) is the main memory of the computer. This memory stores the number codes of the y and S registers of all integrators as well as the codes from the Input and Conversion Unit. Memory locations are also used as registers, pulse counters and for storage of other data (e.g., storage of the data from instruments and transducers, data concerning the limits of change in variables, initial data, etc). The MIM is not directly connected to the integrating unit but through high-speed intermediate or buffer registers. The bits of a word stored in the MIM are transferred in parallel with a frequency of 50-100 kc to the buffer register, from which they are transferred in sequence with a frequency of 1-2 Mc to the adder of the integrating unit. Thus, the MIM is a large "reservoir" from which the codes are transferred to a high-speed internal storage of small capacity using electronic registers or delay lines.

The control unit of the computer delivers bit-number and integrator-number pulses as well as program pulses for all units of the computer including the Input and Conversion Unit, the Function Input Unit and the Output Unit. The Control Unit consists of counters for generating bit number and integrator number pulses. The bit number counter generates bit number pulses $P_1, P_2, \ldots P_n$ through intermediate power amplifiers. By means of power amplifiers the two integrator number counters feed a diode-transformer matrix which has separate output lines for all integrator numbers. The Control Unit also includes program sections which deliver 4-5 programs for interconnection of integrators and insertion of initial data in accordance with the problem to be solved. The programs, the so-called "permanent" memory of the computer, are entered

on punched cards or on ferrite cores with the possibility of switching them according to the problem to be solved. Information from the program sections is read out serially in the form of pulses at a frequency of about 1 Mc by the bit number pulses and integrator number pulses of the control unit. The final result of computation in the integrating unit may be obtained directly at the output in the form of number codes and entered on punched tape.

For automatic control, the computed value A is compared in the machine with the controlled value B of this quantity. The comparison gives an error signal ($\Delta = A - B$) for controlling an external motor. This comparison is performed in the computer itself, and the computer delivers at its output the error signal Δ in code form.

The Output Unit converts the error signal into an electrical voltage for each of the output channels. High accuracy in conversion is not required, as an error of several percent can be tolerated. The error voltage controls motors which in turn control the quantities associated with the process. The angular position of the motor shaft is measured by means of a coded disk or a pulse transmitter. The values of the controlled quantities are fed back in the form of codes (or pulses) for comparison in the integrating unit of the computer. This closes the feedback loop of the computer-controlled system.

The input quantity as well as the controlled quantities at the computer output may be measured by means of transducers whose readings may be inserted into the computer in digital code or converted to digital form by an appropriate converter.

2. GENERAL CHARACTERISTICS OF COMPUTER

Digital integrating computers (digital differential analyzers) may use magnetic drums as internal storages. The use of drum storage usually limits the operating speed of the computer to a clock-pulse (CP) rate of 50 kc. The solid-state digital integrating computer described below has considerably higher operating speed with a clock-pulse rate of 1-2 Mc (i.e., approximately 20-40 times faster than those machines of this type using magnetic drum). The internal memory in this machine is a magnetic core storage (see Chapter IV, Section 4) with semiconductor diodes and transistors.

The use of a magnetic internal memory (MIM) with ferrite cores instead of a magnetic drum is advantageous in that the former has no moving parts; this in turn improves the operational reliability and increases the speed of the computer. The author has described [3] a digital integrating computer in which the internal storage employs solid ultrasonic delay lines; this memory gives a high operating speed. However, the storage capacity of ultrasonic delay lines is limited by the length of the line and is usually not more than 1000 bits. In addition, ultrasonic delay lines possess lower reliability of operation than a MIM and require high amplification of extremely small read-out signals.

In order to solve a given mathematical problem in a DDA provision must be made for the required "capacity" (number of integrators) of the computer. In the serial mode of integration, the time required for execution of one integration step in each of the integrators is determined by the number of integrators and the clock rate of the computer.

In order to increase the machine computing rate, the entire computation process is divided into two or more parallel computing channels; in each of these channels a portion of the problem is solved. For example, in the solution of a point-of-impact problem, one channel may be used to compute the predicted coordinates of the moving object and a second channel may be used to compute its current coordinates and to obtain the final results of computation by comparing the predicted with the current coordinates. For more complex problems it is possible to use more parallel channels.

However, an increase in the number of parallel channels of computation necessitates a substantial increase in the amount of electronic equipment. For two parallel channels, the amount of electronic equipment is only slightly larger than that required for a single channel since the same control unit is used for both channels. Moreover, it must be kept in mind that the total memory capacity remains unchanged for one computing channel and for two parallel computing channels (with the same number of integrators). Since the control unit and the internal memory account for most of the DDA equipment, the above considerations permit us to conclude that it is desirable to use two channels in order to increase the computing rate by a factor of 2.

A somewhat more economical method of increasing the rate of the DDA lies in increasing the clock rate by using computer elements with a higher operating speed. It is entirely feasible at present to have a CP rate of 1-2 Mc by using currently available transistors.

The maximum time required for build-up of initial values corresponding to the true values of the measured parameters may be considerably decreased by selecting initial values closest to the true values. For this purpose, several initial conditions must be stored in the permanent memory. The required values of initial conditions are automatically switched by means of a decoder matrix according to the value of the parameter under measurement, as was shown in Chapter I, Section 9. In addition, for input parameters having a rapid variation or varying over a wider range, a larger number of initial-condition sets are taken than for input parameters having a slow variation or varying over a smaller range.

3. INTEGRATING UNIT

All the computations required for problem solution are performed in the integrating unit. This reduces to successive

integration of the various integrands for the given or computed independent variables. A single integration step for all integrators in a problem solution constitutes a single cycle of problem solution. Duration of the cycle depends on the number of integrators, the number of bits in the integrators, and the clock pulse rate.

As was shown in Chapter I, the integration operation in each integrator consists of two steps. The first step obtains the new ordinate y^* by algebraic summation of the $\Sigma\Delta y$ increments to the current ordinate y, i.e.,

$$y^* = y \pm \sum \Delta y. \tag{1}$$

The second step, the integration operation proper, accumulates the sum of the ordinate by adding the new ordinate y^* to the existing value of integral S, i.e.,

$$S^* = S \pm y^*. \tag{2}$$

Both of these summing operations are performed simultaneously in the machine by means of two single-bit serial adders. Each number y or S is represented by a 16-bit binary code: 15 number bits and 1 sign bit. A positive number is represented by code 1 in the sign bit and a negative number by code 0. In addition, the y and S registers contain one bit in front of the number and one bit after the sign bit, corresponding to the last position P_n in the given cycle (at which the overflow code is obtained). Thus, there are 18 bit positions in the y and S registers, the two extreme positions being used for control of the operations.

The block diagram of the integrating unit is shown in Fig. 195. This unit is connected to a magnetic core storage from which the codes of numbers y and S arrive in sequence and in which the new computed values of y^* and S^* are stored. The increments of the integrand Δy and of independent variable Δx are obtained from the line storing the ΔS overflows in the memory. The ΔS line holds the overflow codes of all integrators. The ΔS overflow code obtained as a result of integration in the given integrator is again stored in the ΔS overflow line to replace the old overflow code for this integrator number. Only selected bits of the entire overflow codes are applied to the Δy or Δx input of the next integrator. Selection of overflow bits at the Δy input of the integrator is achieved by coincidence gate B_1. All the overflow code pulses are applied in sequence to one input of this gate and the time sequence of A_y code pulses from the computer control unit is applied to the other input. This latter sequence selects the overflow pulses of those integrators which, according to the problem solution, must be applied to the Δy input of the given integrator. The code pulses selected by means of gate B_1 arrive in sequence at the input of the $\sum\Delta y$ reversible counter, where they are algebraically summed, forming the code of the incremented sum $\sum\Delta y$. In

summation, code 1 arrives at the add input of the counter as +1, while code 0 arrives at the subtract input as -1. At the beginning of each integration operation, this summed code y^* is transferred from the counter to the $\sum\Delta y$ register and is then serially (bit by bit) transferred from the register to the $\sum_1$ adder where this code y^* is added to the code of the y integrand and gives the sum $y \pm \sum\Delta y$. For addition, the direct code of $\sum\Delta y$ is added in the register and, for subtraction, the 1's complement of $\sum\Delta y$ is added. The resulting new integrand is transferred to another adder $\sum_2$ for integration and stored in the appropriate location of the internal memory to replace the previous code y. Transfer of code y^* is achieved through anticoincidence gate B_2, which passes the codes of all bits (including the sign bit), and then (after the sign bit) the so-called initial or start pulse H is applied to B_2. Gate B_7 passes the clock pulses if the code of the given bit of number y^* is 0 and inhibits passage of the clock pulses if the code is 1. The 1's complement of number y^* appears at the output of gate B_7, with the result that the code pulses arriving at the input of gate B_7 from gate B_6 are inhibit pulses. In order to obtain the 2's complement of number y^*, it is necessary to add 1 in the least significant position of the 1's complement code. This is achieved by gate B_8 in passing to the $\sum_2$ adder's third input the initial pulse H, which is added as +1 to the least significant bit of y^*.

The result of the addition $S = S + y^*$ is transferred from the output of the $\sum_2$ adder via gate B_9 to the S register in the internal memory. This transfers to the S register all the bits of number S^* (including the sign bit) except the last bit, P_n, which is always rewritten as one by means of gate B_{10}. The last bit (P_n) of the number code S serves for inversion of the ΔS increment code. If during the P_n bit the code is 0, the ΔS overflow pulses arrive without change. If during the P_n bit the code is 1, then, as was shown in Chapter I, Section 5, the overflow pulse codes are complemented; this is equivalent to sign reversal of the ΔS increments. The ΔS overflow code pulses from the $\sum_2$ adder output are stored in the electromagnetic overflow storage line via gate B_{11}, which passes only the last (P_n) overflow pulse.

Selection of overflow bits at the Δx (independent variable) input of the integrator is achieved by coincidence gate B_3, which passes only one code pulse chosen according to the A_x program from the control unit. The code pulse selected from the ΔS line is stored in dynamic flipflop FF_1 and at the beginning of each integration operation is transferred (via gate B_4) for storage in flipflop FF_2, freeing flipflop FF_1 to receive code pulse for the next integrator. Flipflop FF_1 is cleared by the first pulse P_1 after the code stored therein has been transferred to flipflop FF_2. Flipflop

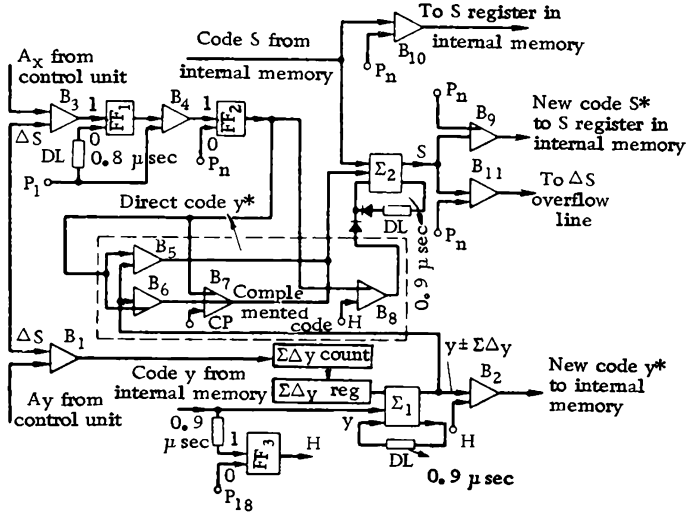

To S register in internal memory
Code S from internal memory
A_x from control unit
New code S* to S register in internal memory
To ΔS overflow line
Direct code y*
Complemented code
New code y* to internal memory
Code y from internal memory
Ay from control unit
ΣΔy count
ΣΔy reg
0.8 μsec
0.9 μsec

Fig. 195. Block diagram of Integrating Unit.

FF_2 is cleared by the last pulse P_n after each step of integration. The state of flipflop FF_2 stores the Δx increment, being positive or negative (corresponding to code 1 or 0). If flipflop FF_2 stores code 1 (positive Δx), the direct code of the new ordinate y^* must be applied to the $\sum_2$ adder (for addition). If flipflop FF_2 stores code 0 (negative Δx), the complement code must be applied to the adder (for subtraction).

The device for transfer of the y^* code to the $\sum_2$ adder in direct or complement code (indicated in the figure within the dashed rectangle) consists of coincidence and anticoincidence gates B_5 - B_8. The direct code of y^* passes through coincidence gate B_5 if the output of FF_2 is voltage pulses (code 1). The complement code of y^* is formed by gates B_6 and B_7. Gate B_6 passes the y^* code to the output only when flipflop FF_2 has code 0; for this purpose the output voltage of flipflop FF_2 is an inhibit voltage in gate B_6. The initial pulse H is used for control of operation of the adders, registers, counter, etc. and is obtained from flipflop FF_3, which is switched to the 1 state approximately 1 μsec after arrival of the start pulse in the y register. Dynamic flipflop FF_3 generates pulses (coinciding in phase with CP) until it is cleared by the last pulse P_n of each integrator.

Depending on the scales used in the problem solution, the initial pulse may be written not only before the least significant bit of the y code but also before the second or a later bit. Beginning with the bit before which the initial pulse is written, summation of the y and $\sum\Delta y$ codes as well as of y and S occurs. For

example, if the initial pulse H is written before the fourth bit in the y code, each of the overflow pulses in the $\sum\Delta y$ counter corresponds to 2^4 units of the y code. In this case, flipflop FF_3 conducts beginning with the fourth bit and triggers the $\sum_1$ and $\sum_2$ adders.

A more complete diagram of the serial integrating unit is shown in Fig. 196. There are two magnetic internal memories for storage of the S codes and the y integrand codes. Each plate of the memory is a ferrite matrix consisting of 64 horizontal rows of words (integrators), each with 18 bit positions. The MIM control unit permits selection of any of the 64 words for read and write operations. The selected work, with all bits in parallel, is read from or written into the read-write buffer register at a frequency of 50 kc.

From the read register, the code pulses arrive at the integrating unit at a frequency of 1 Mc. From the $\sum_1$ and $\sum_2$ adders, the y and S codes arrive at the y and S write buffer registers of the MIM at a frequency of 1 Mc. Thus, the read and write buffer registers serve the purpose of converting the parallel code arriving from the MIM with a frequency of 50 kc to a serial code with a frequency of 1 Mc.

An electromagnetic delay line DL_1 stores the overflow pulse codes from the output of the $\sum_2$ adder. It consists of 17 line segments with a delay of 1 μsec each. The 17 overflow pulses for the 17 integrators are stored in sequence in DL_1 and are rewritten after each integration step by anticoincidence gate B_{12}. This gate passes for rewrite all pulses stored in the line except the last pulse, which (during passage of the last overflow pulse P_{18}) is replaced by a new overflow code from an integrator. In this case, B_{12} is cut off and a new overflow pulse arrives at the overflow line via gate B_{11} from the $\sum_2$ adder output and is stored in the line instead of the previous overflow pulse. Since the length of overflow line DL_1 is one bit position shorter than the number of bits in each integrator, the overflow pulses are displaced by one bit position upon each rewrite of the codes in DL_1. This permits writing the overflow pulses in numerical order of the integrators from which they are obtained. Selection of the overflow pulse codes stored in delay line DL_1 is achieved by means of code-select circuits according to the A_x or A_y programs arriving from the control unit of the computer, as was discussed above.

Problem solution often necessitates feedback between integrators or the interconnection of integrators which are some cycles apart. This requires storage of the overflow pulses for a relatively long time (as long as the next integration step). Such storage is achieved by means of a second electromagnetic delay line DL_2 consisting of 18 line segments with 1-μsec delay each. The overflow pulses from any of the 18 integrators may be stored in this line for the duration of one integration step. The sequence of writing and reading codes from DL_2 is controlled by read-write gates

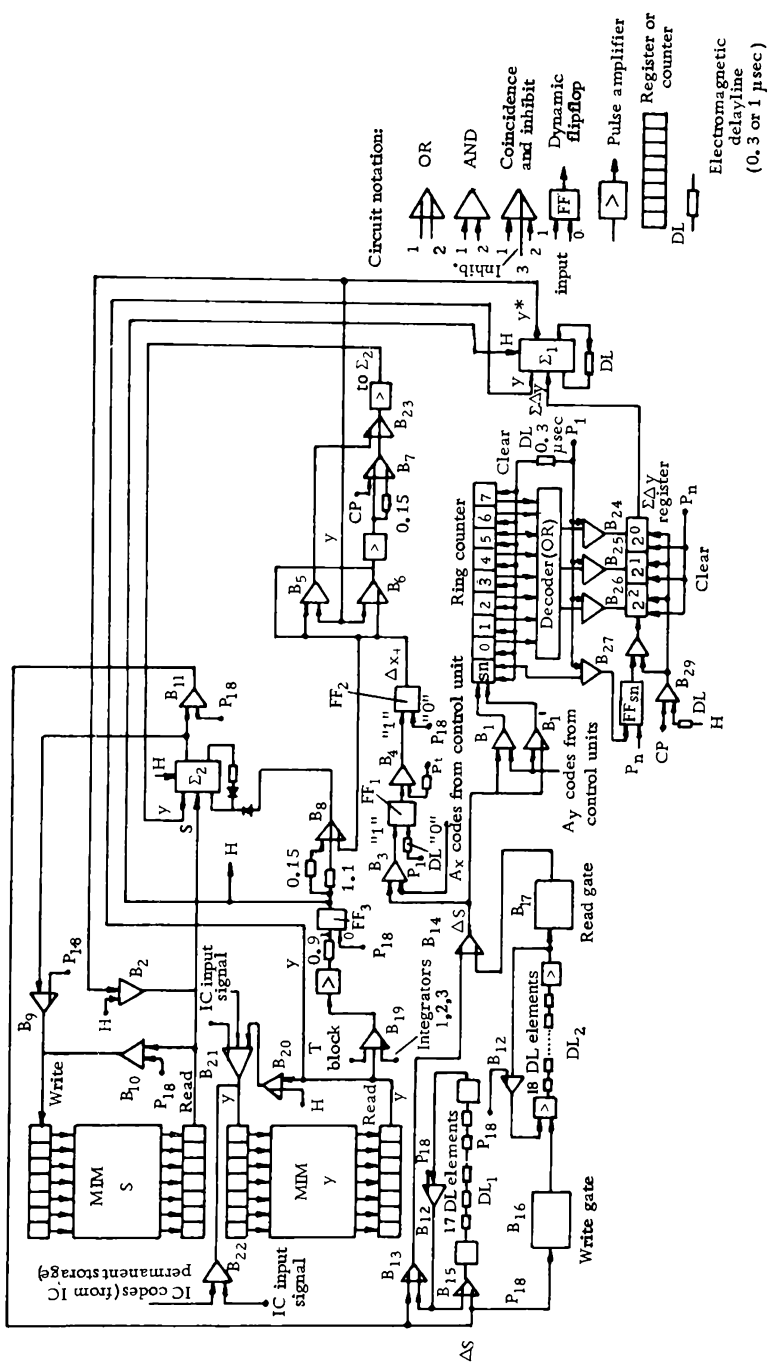


Fig. 196. Diagram of serial integrating unit.

(see Section 9) which determine the storage locations of the overflow pulses and their destinations. At appropriate moments in time, the required overflow pulses are selected from this line and directed to the inputs of other integrators.

The codes selected from the ΔS overflow line are applied to the Δy or Δx input of the integrators, as was discussed above. The overflow codes from the ΔS line arrive at the Δy input of the integrators through gates B_1 (code 1) and B_1' (code 0). In this case, code 1 corresponds to addition (+1) to the $\sum \Delta y$ counter, while code 0 corresponds to subtraction (−1). The $\sum \Delta y$ counter here is a reversible ring counter (see Chapter III, Section 14) with dynamic flipflops which counts up to seven and has a sign flipflop. The counter has two inputs (add and subtract) and may count pulses with a repetition frequency of 1 Mc. The counter state corresponding to the algebraic sum of the stored pulses is converted by a simple diode decoder to the $\sum \Delta y$ number code with three number bits and one sign bit. The resulting $\sum \Delta y$ number code is transferred through gates B_{24}-B_{27} in parallel to the $\sum \Delta y$ register (with dynamic flipflops). Gates B_{24}-B_{27} are caused to conduct by the start pulse P_1; this same pulse, going through the 0.3-μsec delay line DL, clears the counter to the zero state. The sign bit of the counter is stored in dynamic flipflop FF_{sn}. The output of this flipflop is connected via gate B_{28} to the other bit positions of the $\sum \Delta y$ number code in the register. As a result, the sign bit, in transferring the $\sum \Delta y$ number code from the register to the $\sum_1$ adder, is repeated at the output of the register as many times as the number of remaining bits in the y code after the $\sum \Delta y$ code.

Shift pulses are applied to the register from the clock pulse generator through gate B_{29}, which passes the pulses for all bits of each integrator. The $\sum \Delta y$ code stored in the register is transferred by a series of shift pulses to the $\sum_1$ adder input. The direct code is transferred for a positive number $\sum \Delta y$ and the complement code for a negative number. The complement code is obtained directly by means of the counter and the $\sum \Delta y$ register.

With the exception of certain additional details, the remaining elements of this integrating unit correspond exactly to those of the previously discussed unit. In order to increase the operating reliability of anticoincidence gates B_7, B_8, and the like, the inhibit pulse is stretched by means of small delay lines. Reversal of the output signal polarity (inversion) is achieved by means of transformers not shown in the diagram. All coincidence and anticoincidence circuits as well as OR circuits use semiconductor diodes.

Amplification of pulses from the logic circuits as well as at the output of the electromagnetic delay lines is achieved by means of transformer-coupled transistor circuits. All elements of the integrating unit with clock rate of 1 Mc use transistors of types P402-P502 (see Chapter III).

4. CONTROL UNIT

The control unit of a digital integrating computer must achieve control of the entire computing process and of the insertion of initial input data (the initial values of the problem solution program and the initial conditions). For this purpose the control unit shown in Fig. 197 contains the following elements:

a) A bit counter for counting the 18-20 bits of a number code (pulses P_1, P_2, $P_3, \ldots, P_{20}$), having 20 bit lines as outputs.

b) An integrator number counter for counting the 16 integrators, having 16 integrator number lines as outputs.

c) An integrator group counter for counting four groups; each group has 16 integrators (a total of 64 integrators).

d) A program storage circuit providing the timing sequence for coded programs of A_x, A_{y1}, A_{y2}, A_{y3}, and IC (initial conditions). The coded programs for A_x, A_{y1}, A_{y2} and A_{y3} serve to select overflow pulses from the overflow increment storage lines at the input of the integrators and to store the overflow increments at the proper locations. Programs A_x and A_y interconnect the integrators in accordance with the setup diagram of the given problem. The IC program inserts initial values in the y and S registers of all integrators. Insertion of the input channel increments which are stored in lines or registers is achieved by means of the same programs. The A_{y3} program stores increments at the appropriate locations of the increment storage lines for their prolonged storage (more than one integration step).

e) Pulse generating circuits.

For a clock pulse frequency of 1 Mc, the ring counter consists of 20 stages, each containing a high-power dynamic flipflop and a gate amplifier (see Chapter II). The output winding of the power amplifier transformer drives the output line whose corresponding dynamic flipflop is in the 1 state. High-power pulses from the CP and CPS (clock pulse shift) generator (frequency 1 Mc) are applied to the counter input. After each 20 CP pulses, a single pulse is obtained and applied to the input of the integrator number counter.

The integrator counter consists of two series-connected ring counters using dynamic flipflops; one counter has 16 stages and the other four. The first counter counts integrator numbers up to 16 and receives pulses from the bit counter at a frequency of 1000/20 = 50 kc, while the second counter receives pulses from the first counter at a frequency of 50/16 = 3.1 kc. The output transformers of the power amplifiers of the integrator number counter drive

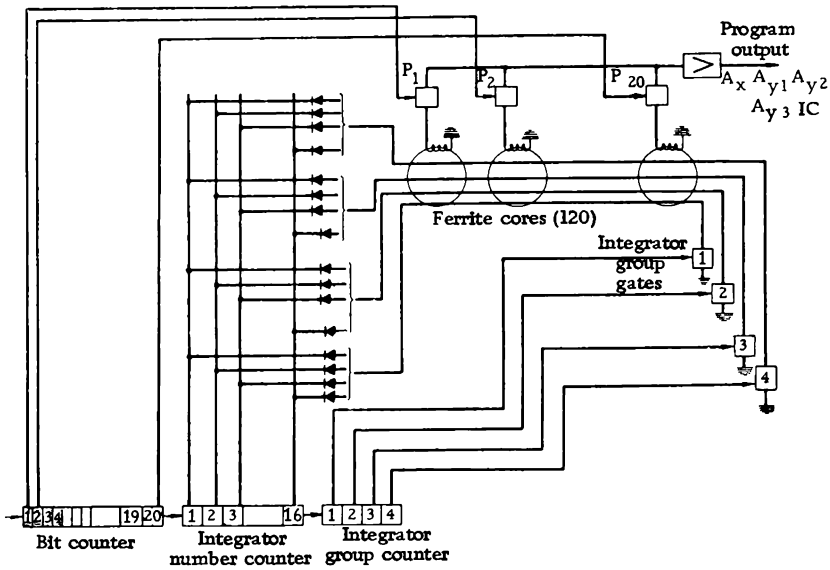


Fig. 197. Block diagram of control unit.

the output line running through a row of ferrite cores; the coded program is achieved by threading of these lines through the cores. For each of the A_x, A_{y1}, A_{y2}, A_{y3} and IC programs, 20 cores (according to the number of bits, including the control bits) are threaded. In addition, several cores are reserved for the machine instructions (pulse counting, integration, insertion of functions, servo integrator, data input, etc). Thus, approximately 120 cores are threaded in sequence. The number of conductors threaded through the cores and connected in series with diodes is equal to the number of integrators groups (i.e., four groups of conductors with 16 lines each), each group being connected to the ground through one of gates B_1-B_4.

The timing sequence of pulses for each program is taken from the output windings of the ferrite core transformers, which are in themselves a type of "permanent memory" for storage of programs for interconnection of the integrators according to the problem solution scheme.

Each bit position of one program code has a corresponding ferrite core transformer. Storage of 1 or 0 at the bit position is achieved, respectively, by threading or not threading through the corresponding core transformer. In reading out the stored information, an emf is induced in the output winding of the transformer if a line is threaded through the core.

Thus, for each integrator number, a parallel 20-bit code for the given program appears at the output of the transformers. By means of coincidence gates B_1-B_{20}, controlled by the bit line numbers P_1-P_{20}, this code is converted to a serial code (i.e., a time sequence

of pulses) for each integrator number. Time sequence A_x for each integrator has code 1 in only one bit position. Time sequence A_y is similarly converted, but here for several integrator lines up to seven windings may be connected in different bit positions (corresponding to the seven inputs for Δy increments).

The pulse sequence for inserting initial conditions IC in the y register may have any y_0 number code for each integrator. It is formed in a manner similar to that for the A_x and A_y programs. In order to reduce the problem solution time, it is often necessary to switch initial conditions (automatically or manually) during the processing of initial data by the machine. This may be achieved by means of a switch. For each sequence of initial conditions, there is a separate row of cores or there are additional windings on the existing cores; these additional windings are switch-selected according to the required initial conditions.

In addition, the control unit has start-stop provisions. The computer is started by means of a start flipflop and stopped by a halt flipflop; the former controls a gate delivering the initial (start) pulse to the integrating unit and also connects other auxiliary circuits. Upon starting the computer, a single pulse IP from a special IP generator (for example, a multivibrator) is sent to the counters of the control unit. Halting of the computer at any given integrator and at a given integration step provides sufficient time for print-out of the computation results. In order to stop the computer at a given step, the halt flipflop must be set to the 0 state.

In addition to the above, it is necessary to perform the following operations in order to stop the computer at a given integrator for a given cycle:

a) Inhibit count of the integrator numbers; this requires that the input circuit of the integrator counter be cut off.

b) Inhibit application of the read-write signals to the magnetic memory.

c) Inhibit the read and write operations for the ΔS overflow pulses in the overflow storage lines.

d) Retain the accumulated $\Sigma \Delta y$ pulses in the counter; this requires inhibition of resetting or clearing of the counter by pulse P_1.

e) During the machine stop, clearing of the y and S write registers of the selected integrator in the magnetic memory is inhibited.

5. PROGRAM CODE GENERATION UNIT

For the solution of various problems the digital integrating computer must have a suitable device for providing integrator interconnection programs (programs A_x, A_{y1}, A_{y2}; program A_{y3} will not be discussed here) and for inserting initial conditions. The scheme for program code generation shown in Fig. 198 consists of a punched card PC on which there are 18 rows of hole positions (corresponding to 18 bit pulses) and 64 columns of hole positions (corresponding to 64 integrator numbers). The presence or absence

of a hole at a position on the card corresponds to code 1 or 0, respectively. There are 18 ferrite core transformers for 18 bit positions. There are 64 windings on each core, with one terminal of each winding connected to the pulse source and the other terminal connected to the 64 integrator number lines. There is one output winding on each core. These output windings are connected through gates B_1-B_{18} (controlled by bit pulses P_1-P_{18}) to a common line from which the program is obtained in the form of coded sequences of pulses.

The scheme for sequencing the 64 integrators is shown in the block diagram of Fig. 199 and consists of a bit counter, one integrator number counter (in two sections) and a diode-transformer matrix. The 64 output lines of the matrix drive the windings of the cores in Figure 198. The ring bit counter has 18 stages for the 18 bits. Clock pulses (frequency 1 Mc) are applied to the bit counter input. There is a power amplifier at each of 18 output lines of the bit counter. These lines are the bit lines of the computer. After receiving 18 CP pulses, the bit counter gives a single pulse output which is applied to the input of the integrator number counter.

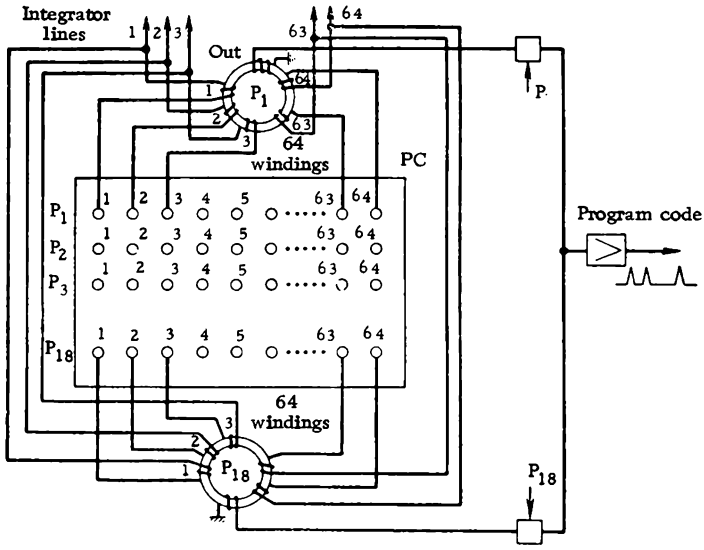


Fig. 198. Punched cord program unit (A_x, A_y, A_{y_2}, IC).

The integrator number counter consists of two series-connected ring counters with 8 stages each. They count the integrator numbers up to 64. Through power amplifiers the output windings of the counter stages drive the horizontal and vertical lines of the diode-transformer matrix, which has 64 output lines corresponding to the 64 integrator numbers. These lines feed cores P_1-P_{18} of each program unit. There is a total of four programs: (a) A_x for connection of the x inputs of the integrator; (b) A_{y1} for selection of the

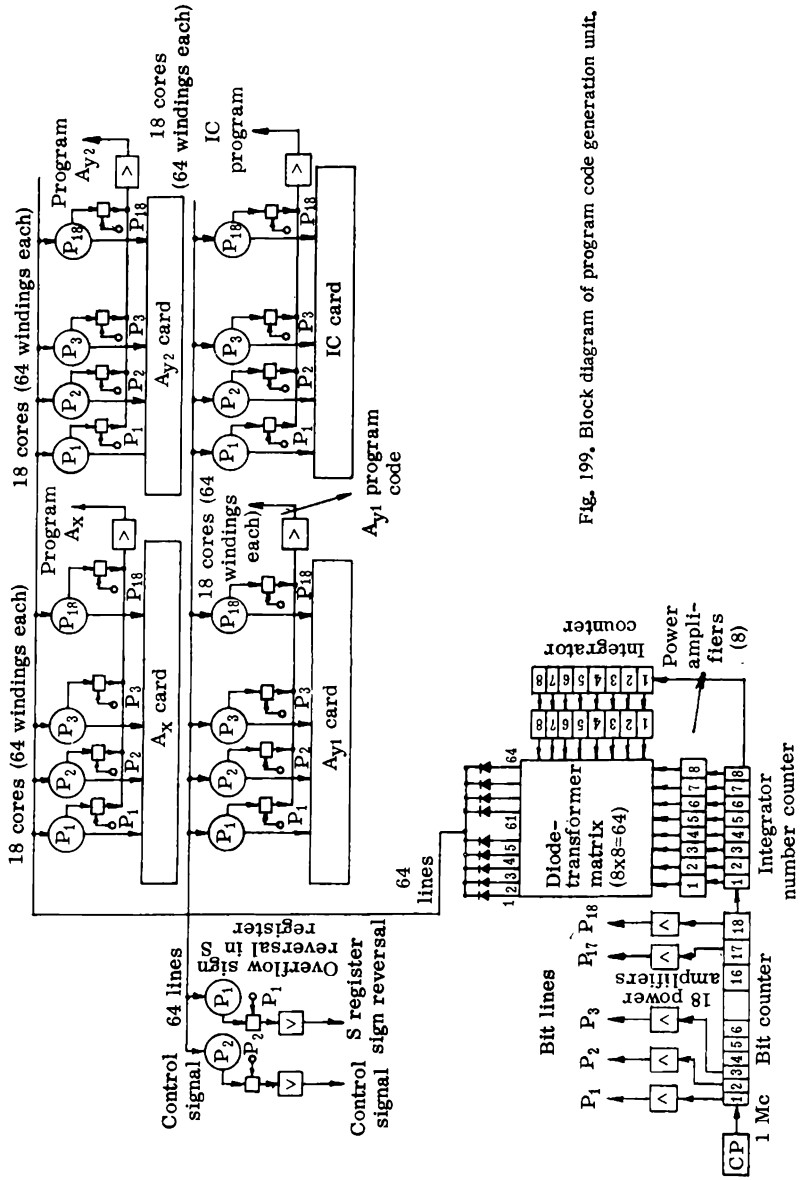


Fig. 199. Block diagram of program code generation unit.

increment from the current overflow line at the Δy inputs of the integrators; (c) A_{y_2} for selection of increments from the long-term overflow storage line and of increments from the storage line for input channels; (d) insertion of the initial conditions code in the y register. Each of these programs is stored on a punched card. Readout is achieved serially from the output windings of the ferrite cores through gates controlled by the bit pulses P_1-P_{18}. Each of the cores has 64 windings which are driven by the 64 integrator lines (sequentially in time) and are connected to form current loops through holes in punched cards. For code entry, the P_1-P_2 bits (see Fig. 199) of the S register have two separate cores, each with 64 windings fed from the 64 integrator number lines. The first core P_1 serves for sign reversal of the ΔS overflow at the integrator output and the second core P_2 is for control purposes. The remaining codes of the S register are entered as code 0 in all bit positions (except P_1, P_2 and the sign bit). The punched card for each of the four programs has 18 rows (codes) corresponding to the 18 bit positions. In each row, a punched hole at a bit position codes the selection of the overflow pulse. There are $18 \times 64 = 1152$ bit position on each card for the 64 integrators.

The windings of the ferrite cores for each program unit are driven by the integrator number lines and are connected serially or in parallel. In serial connection, the windings of the 18 cores are threaded by a single conductor from each integrator number line. Thus, each integrator number line must drive four program units; each unit contains 18 cores. Since the output windings of the ferrite core transformers deliver very little power (which is subsequently amplified), a relatively low current is drawn from the integrator number drive lines.

To change a problem solution, only the four punched cards for programs A_x, A_{y_1}, A_{y_2} and IC as well as the code card for overflow sign reversal in the S register are replaced. If the computer must be automatically switched from the solution of one problem to that of another, it is necessary to have two (or more) program code generation units for two (or more) programs. These units are switched by means of gates.

6. INCREMENT INPUT UNIT

In the digital integrating computer, the unit for inserting input increments (in digital form) removes the data from the converter for each input and transfers an increment of this input to the computer (Fig. 200). The converters are binary coded discs mounted on a shaft for coarse and fine reading. Ambiguity in reading codes at certain brush positions on the binary-coded disc is eliminated by means of a simple transistor circuit (see Chapter VII, Section 6). In Fig. 200, the binary-coded discs with brushes are represented by 16-bit registers, one for each input channel (there are three channels in this figure).

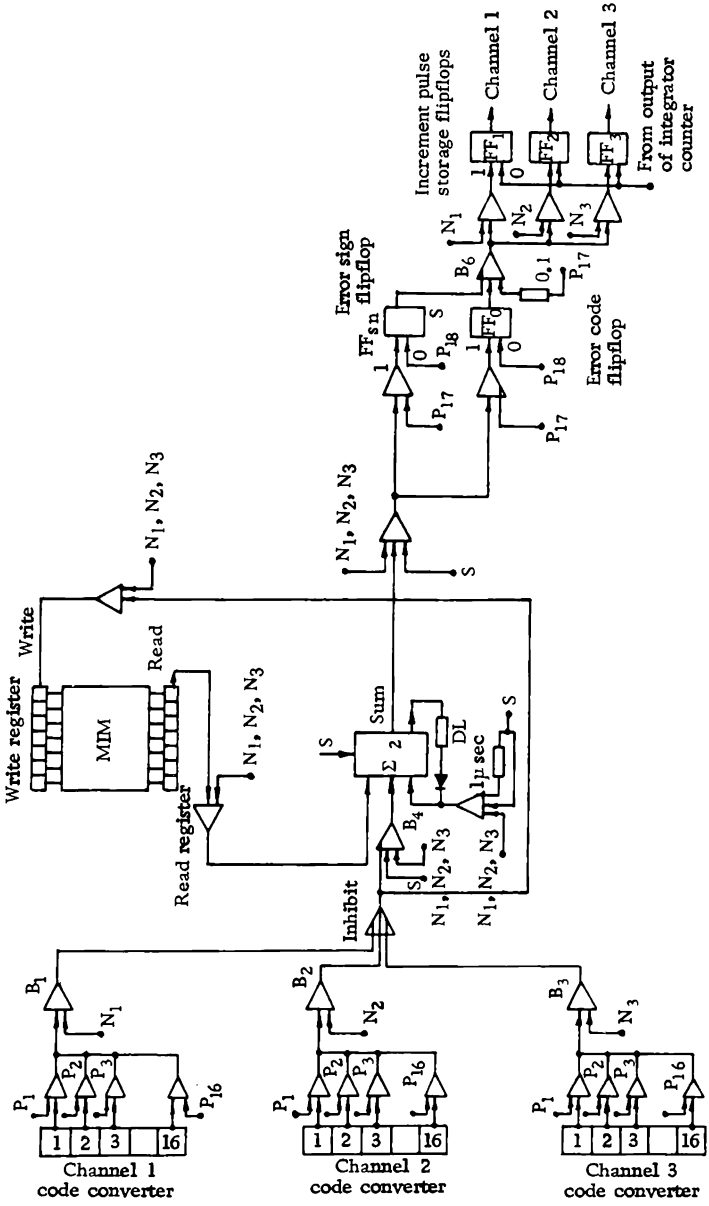


Fig. 200. Initial-condition input unit.

By means of gates controlled by the P_1-P_{16} pulses (from the control unit), the parallel code of the binary-coded disc is converted to a serial code at a pulse repetition frequency of 1 Mc. The code pulses are serially read out from each channel and written into special locations (1, 2, 3) of the internal memory. The write operation occurs at the appropriate moment through gates B_1-B_3. These gates are controlled by the lines of integrator numbers 1, 2 and 3, which determines the locations in the internal memory. Simultaneously with the write operation, the previous code A of the given channel arrives at the input of the $\sum_2$ adder where it is compared with the new code $\overline{A}$ (i.e., the subtraction: $A - \overline{A} = \Delta$). The sign of the Δ error code subsequently serves for control of the gates through which the increment code pulse for the input variable is inserted into the machine. For the purpose of subtraction, the 2's complement of code $\overline{A}$ from the converter is applied to the adder input by means of gate B_4, which also performs the 1's complement of code $\overline{A}$. The addition of $+1$ in order to obtain the 2's complement is achieved at the third input of the adder via gate B_5 through which the start pulse S passes and is added as $+1$ to the least significant bit.

Comparison of the previous value of the A code with its new value for each channel is achieved once each integration step. The resulting Δ error code is stored in flipflop FF_0 (code 1 in any of the number bits except the sign bit). The sign of the error code is stored in flipflop FF_{sn}, which inhibits passage of the error code pulse through gate B_6. Code 0 at the output of gate B_6 corresponds to a positive error and code 1 to a negative error. If the error $\Delta = 0$, code 0 occurs at the output of gate B_6. For $\Delta = 0$, there will be a sequence of pulses 0,1, 0,1 etc, (i.e., true zero). The resulting code for the increment of the input value for each channel is stored in dynamic flipflops FF_1-FF_3, which are cleared at the end of each integration step.

7. OUTPUT CONVERTER FOR MOTOR CONTROL

The coupling of a digital computer to a controlled device is achieved by means of a converter at the computer output. The converter must convert the Δ error code obtained from the digital computer to a proportional electrical voltage which controls a servomotor (Fig. 201). One conversion method is described below.

For this purpose the Δ error code at the computer output is read out a certain number of times per second. Upon the read control signal, the error code Δ is serially transmitted to a storage register via gates B_1-B_6. The serial Δ code is converted into a parallel code: gate B_1 passes only the first bit of the code under the control of pulse P_1; gate B_2 passes the second bit under the control of pulse P_2; etc. Pulses P_1, P_2, P_3, etc. (the computer control pulses) are in time coincidence with the passage of the code for the given bit from the computer. Each bit of the Δ code is stored in a dynamic

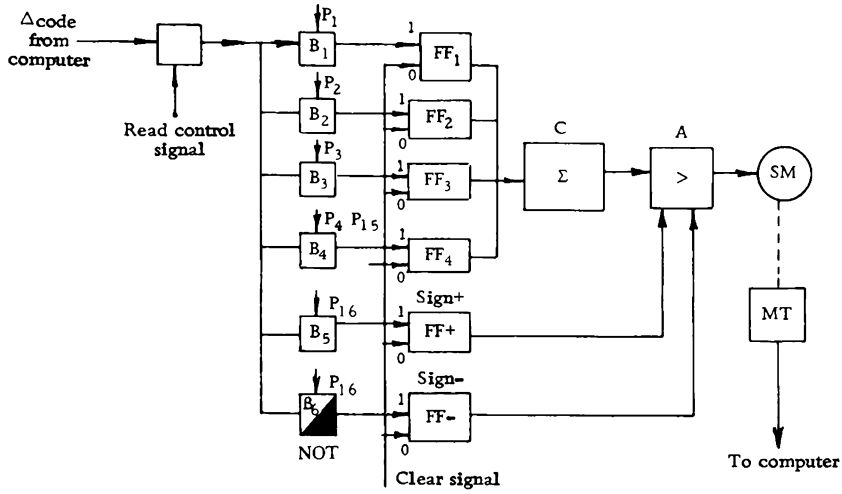


Fig. 201. Computer data output diagram

flipflop. For motor control, only the first three or four digits of the Δ code need be stored, and if any of the remaining significant bits of the Δ code is 1, code 1 is stored in one flipflop. The motor is controlled by making the error voltage within very narrow limits (close to zero) proportional to the motor speed. When the error is large, the motor rate is at its maximum. In view of this, the FF_1-FF_3 flipflops store the codes of only the first three bits of the Δ code, while flipflop FF_4 stores code 1 if any of the remaining bits is 1. The Δ sign code is stored in flipflop FF_+ (code 1) or flipflop FF_- (code 0). In the latter case, the read control signal P_{16} passes through anticoincidence circuit B_6.

Dynamic flipflops FF_1-FF_4, storing the Δ error code, yield at the output a voltage which is proportional to the "weight" of the error bits. For this purpose, for example, the number of turns in the secondary winding of the transformer at the output of each flipflop is chosen proportional to the "weight" of the given digit. The voltages obtained from the output of flipflops FF_1-FF_4 are converted to a d-c voltage and summed in converter C.

Converter C is a circuit in which the voltage of each transformer is rectified by means of diodes, smoothed by a capacitor filter, and serially summed by resistors. By means of a diode-transformer modulator circuit, the resulting d-c voltage is converted to an a-c voltage with a frequency of 400 cps. This a-c voltage is then amplified by amplifier A which controls the servomotor. The direction of motor rotation is determined by the state of the sign flipflop FF_+ or FF_-, i.e., code 1 or 0 in the Δ sign bit. The sign flipflops switch the current direction in amplifier A in such a manner that in the presence of code 1 the motor rotates in one direction and in the presence of code 0 rotates in the other direction. If

the error voltage at the amplifier output is zero, the motor remains still.

The Δ error code must arrive from the computer at a sufficient frequency to ensure continuous control of the motor. In addition, special networks must be provided in the amplifier for differentiation and integration of the error voltage for different modes of motor control.

A transducer MT connected to the shaft of the motor SM permits measurement of a parameter generated by the motor. The output of this transducer is fed back to the computer where the computated quantity is compared with the measured quantity. When they are precisely equal (i.e., $\Delta = 0$) the motor will remain still.

The above described method of obtaining data from the computer is extremely economical and does not require high accuracy in conversion of the output data. Since comparison of numbers is achieved in the computer itself, it requires a Δ error code of only several bits at the computer output for motor control.

8. INCREMENT-PULSE STORAGE

The memory for storing increment pulses, also a part of the internal storage of the computer, is used for the following purposes:

a) As a ΔS line for storage of current overflows (i.e., the overflows from the output of all integrators located one next to the other in a time sequence);

b) For storage of the overflow pulses of those integrators separated in time and of the overflow pulses of those integrators used for feedback. In this case, the storage lasts for one iteration cycle (i.e., for the time of one problem solution);

c) For storage of pulses from external pulse channels;

d) For storage of the increment pulses of input quantities from the analog-to-digital converter.

In the ternary system of coding increment, two storage lines are required for the +1 and -1 overflow increments of the integrators. The absence of increment (zero) corresponds to code 0 in both lines. Fig. 202 shows the block diagram of a delay line for storing current overflows in ternary coding. The ΔS_+ line serves for storage of positive overflows and the ΔS_- line for negative overflows. Rewrite of the codes stored in the line is achieved through gates B_1, which are cut off upon passage of the last pulse P_{18}. At this moment, the new overflow from the output of an appropriate integrator is inserted into the line. For displacement of the code pulses in the overflow line by one bit, the capacity of the ΔS line is one bit less than the number of bits of the register in each integrator. In this case the new overflow code is written into the overflow line instead of the old code, since gates B_1 inhibit rewrite of the latter.

Selection of the overflow pulses from the ΔS line at the Δx and Δy inputs of the integrators is achieved by means of gates B_2 and B_3 (for Δx input) and B_4 and B_5 (for Δy input). The first pair of

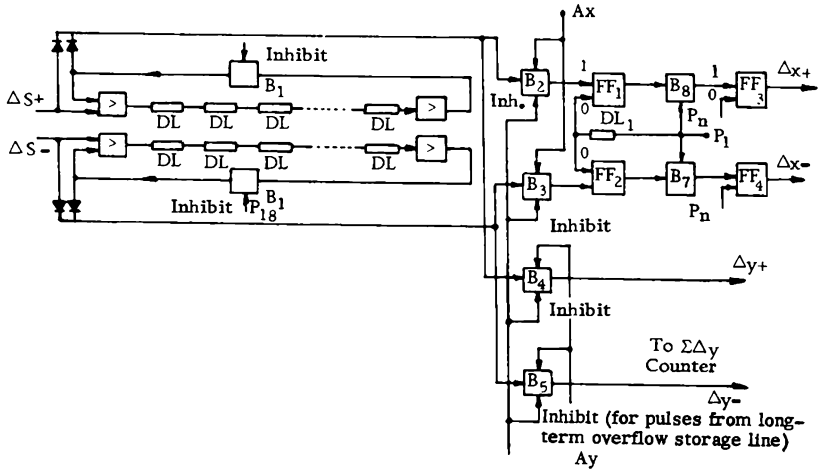


Fig. 202. Delay line for storage of current overflows.

gates conducts according to the signal of the coded A_x program, and the second pair conducts according to the signal of the coded A_y program. Selection of overflow pulses from the line is inhibited if there are pulses which arrive at the Δx or Δy input of the integrators from the long-term overflow storage line. This inhibition is achieved by application of an inhibit pulse to gates B_2-B_5. The code pulses selected from the overflow line are stored in dynamic flipflops FF_1 and FF_2, which control gates B_6 and B_7. Upon application of the first pulse P_1, the codes stored in flipflops FF_1 and FF_2 are transmitted through gates B_6 and B_7 to flipflops FF_3 and FF_4, while flipflops FF_1 and FF_2 are cleared by this same pulse through a small delay line DL_1. This clearing makes flipflops FF_1 and FF_2 again ready for storage of the codes arriving from the overflow line (for the next integrator), while the codes transmitted to flipflops FF_3 and FF_4 are used as the Δx_+ or Δx_- increments in integration. The Δy_+ or Δy_- increments selected from the ΔS line by means of gates B_3 and B_4 are fed directly to the $\sum \Delta y$ increment counter input for addition or subtraction.

For longer storage of the overflow pulses the line delay is made equal to the duration of one iteration cycle (Fig. 203). Here overflow pulses (in ternary coding) are stored in two shift registers R_1 and R_2 for positive and negative increments ΔS_+ and ΔS_-. Overflow pulses are entered into registers R_1 and R_2 through gates B_1 and B_2; these gates conduct only upon the application of the last bit pulse P_n and for those integrators whose outputs are required to be stored for one cycle. The above pulse gating through B_1 and B_2 is controlled by gate B_3. To enter ΔS_+ or ΔS_- pulses into the registers, gates B_4 and B_5 inhibit the pair of overflow pulses which are circulating from the register output to input.

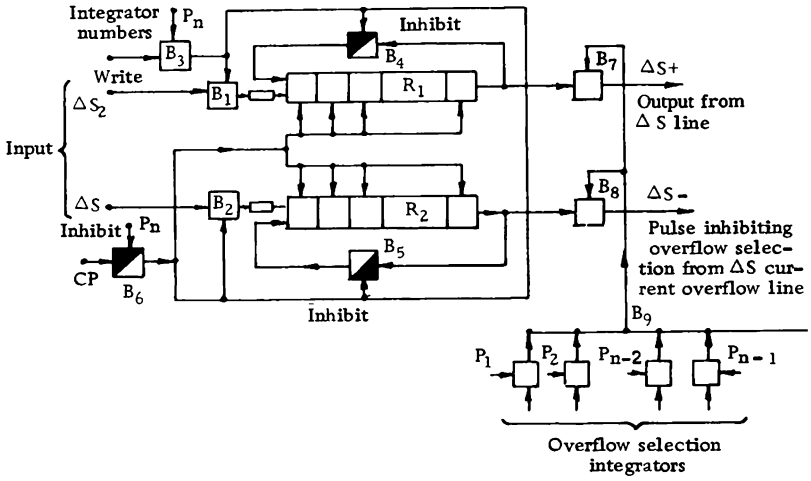


Fig. 203. Single-cycle overflow storage line.

Registers R_1 and R_2 have bit positions one less than the number of bits in the integrators. By means of gate B_6, $n-1$ shift pulses (the last pulse P_n inhibits the shift) are applied to the shift bus. The codes stored in the two registers are normally rewritten in correct sequence without the need of additional shift. When a new overflow pulse is being written, gate B_3 delivers at the arrival of the last bit pulse P_n one more shift pulse to the shift bus of both registers. This pulse shifts the codes in the registers 1 bit to the right and at the same time inhibits (by means of gates B_4 and B_5) rewrite of the codes previously at this location. Hence the code to be entered arrives at a cleared stage of the register. As the register is filled, the code pulses are arranged therein in order of their entry.

Selection of code pulses from the registers at the integrator inputs is achieved by gates B_7 and B_8, which in turn are controlled by gates B_9. Gates B_9 are caused to conduct by the lines of those integrators to the input of which the code pulses stored in the registers are to be applied. Since the order of the code pulses in the registers is known, it suffices to apply for their selection the appropriate pulses of P_1-P_{n-1} to the inputs of gates B_9. The output of gates B_9 may also be used as a signal for inhibiting selection of overflow pulses from the current overflow line.

9. WRITING AND READING OF INCREMENT PULSES

For a relatively large number of integrators, the storage of increment pulses from the integrators and from the external channels requires not one but several parallel increment storage lines. It is necessary to switch these lines for writing and reading in

accordance with the program. As an example, we shall discuss the case in which there are five pulse storage lines, each with two channels for storage of the +1 and −1 increments because of ternary coding. One of these five lines is used for the storage of current overflows; two lines are used for storage of pulses for one cycle; and two lines for storage of the increments of 40 input channels. If there are 20 bit pulses for an 18-bit number (including two control pulses), the length of each overflow line will be designed for 18 code positions.

The y or S register (Fig. 204) contains 20 bit positions P_1-P_{20}. The first two pulses P_1 and P_2 are used for control purposes. In the y register, pulse P_1 may be used for insertion of an external

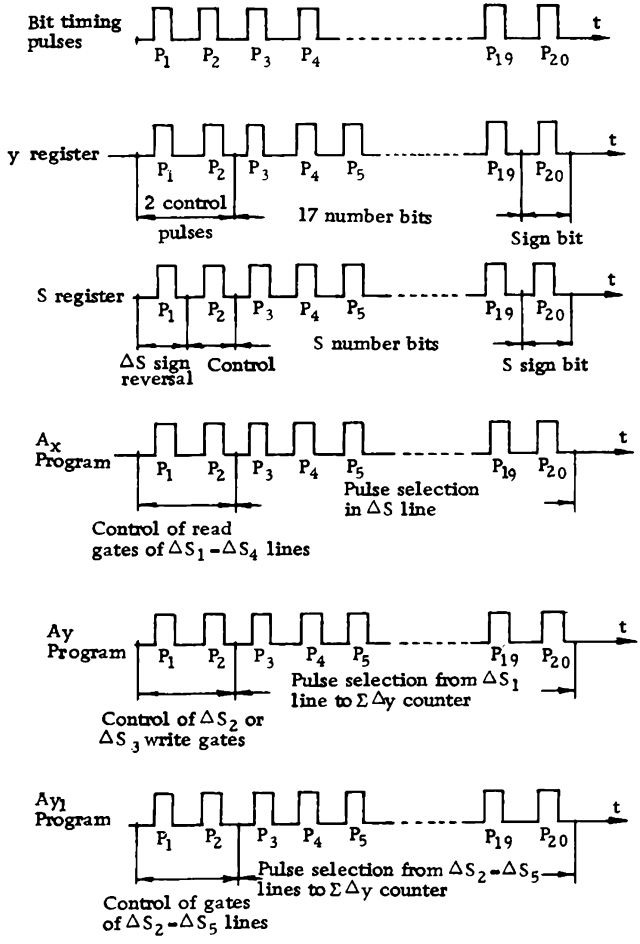

Bit timing pulses
P1 P2 P3 P4 P19 P20 t
y register
P1 P2 P3 P4 P5 P19 P20 t
2 control pulses
17 number bits
Sign bit
S register
P1 P2 P3 P4 P5 P19 P20 t
ΔS sign reversal
Control
S number bits
S sign bit
Ax Program
P1 P2 P3 P4 P5 P19 P20 t
Pulse selection in ΔS line
Control of read gates of ΔS1-ΔS4 lines
Ay Program
P1 P2 P3 P4 P5 P19 P20 t
Pulse selection from ΔS1 line to ΣΔy counter
Control of ΔS2 or ΔS3 write gates
Ay1 Program
P1 P2 P3 P4 P5 P19 P20 t
Pulse selection from ΔS2-ΔS5 lines to ΣΔy counter
Control of gates of ΔS2-ΔS5 lines

Fig. 204. Timing Diagram.

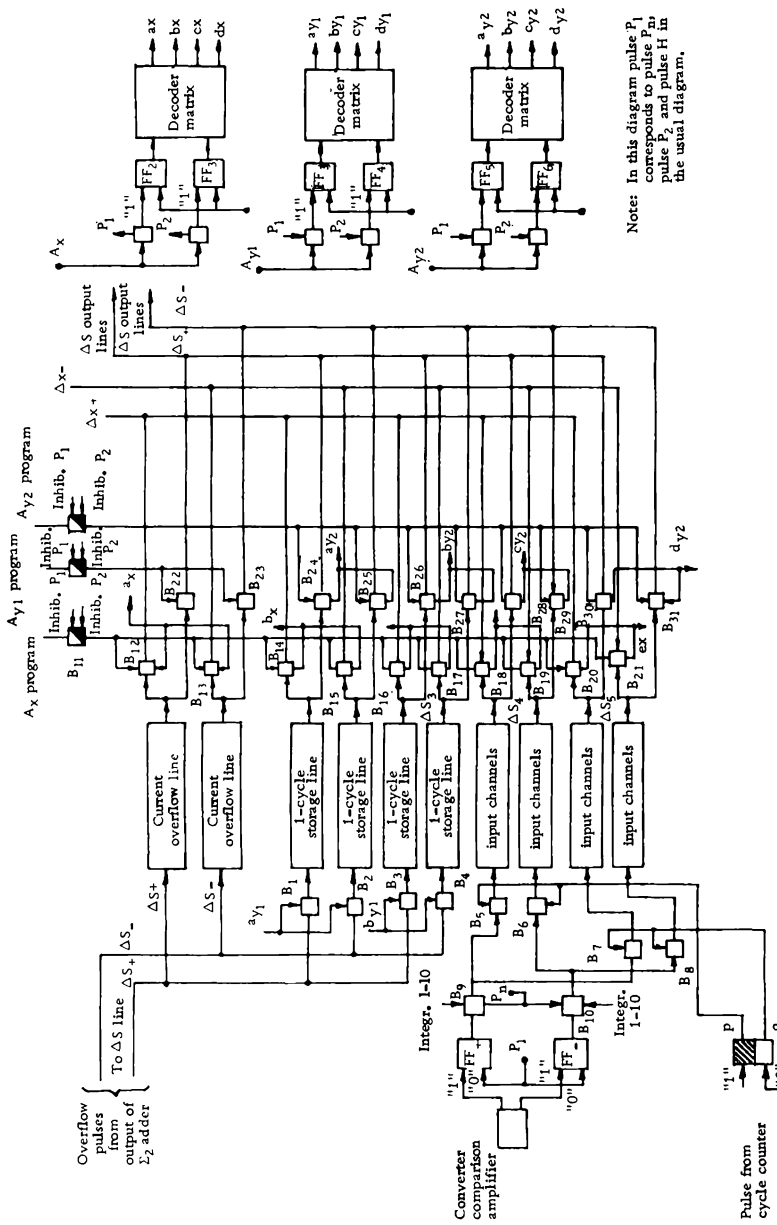

Decoder matrix
Current overflow line
1-cycle storage line
input channels
Integr. 1-10
Converter comparison amplifier
Pulse from cycle counter
Overflow pulses from output of Σ2 adder
To ΔS line
ΔS output lines
Note: In this diagram pulse P1 corresponds to pulse Pn, pulse P2 and pulse H in the usual diagram.

Fig. 205. Increment selections from five storage lines.

number into the y register, while pulse P_2 may be used for delivery of a number from the y register to an external circuit. In the S register, the P_1 pulse is used for sign reversal of the S overflow at the integrator output, while the P_2 pulse triggers the flip-flop which controls data input from external sources. The P_1 and P_2 pulse codes are inserted into the S register upon starting the DDA and remain unchanged during the problem solution. In addition, code 1 is inserted as the initial condition into the most significant bit of the S register; insertion of code 1 corresponds to filling half of maximum capacity of the S register. Such filling of the S register is equivalent to rounding off the number obtained as a result of integration.

The A_x program code, which contains 20 pulse positions, selects the required overflow pulses at the Δx input of the integrator. The first two code pulses P_1 and P_2 serve for connection during readout of one of the available $\Delta S_1 - \Delta S_4$ lines from which will be selected the ΔS overflow at the Δx input of the given integrator. Codes P_1 and P_2 are stored by two flipflops FF_1 and FF_2 (Fig. 205) of the decoder circuit, at the output of which there are four lines a_x, b_x, c_x, d_x controlling the gates switching the $\Delta S_1 - \Delta S_4$ lines. Line e_x (controlling gates $B_{20}-B_{21}$ for readout of codes from the ΔS_5 line) is available from another matrix of the A_y program. There are two programs A_{y1} and A_{y2} for selection of overflow pulses at the Δy input of the integrators. The A_{y1} program serves to select the overflows only from the ΔS_1 current overflow line. The two code pulses P_1 and P_2 in this program are used for control during write-in of the overflow pulses in the ΔS_2 or ΔS_3 storage line (one-cycle storage); and the remaining code pulses for selection of the required ΔS overflow from the ΔS_1 line. Pulses P_1 and P_2 are applied to the input of flipflops FF_3 and FF_4 of the decoder matrix, the output lines of which a_{y1}, b_{y1}, c_{y1}, d_{y1} control write gates B_1 and B_2. The A_{y2} program is for selecting the overflows from the ΔS_2 and ΔS_3 lines (single-cycle storage) or from the ΔS_4 and ΔS_5 lines for storage of the increment pulses from the 40 input channels. The first code pulses P_1 and P_2 of the A_{y2} program select one of the $\Delta S_2 - \Delta S_5$ lines during readout. These pulses are stored by flipflops FF_5 and FF_6 of the decoder circuit. Its four output lines (a_{y2}, b_{y2}, c_{y2}, d_{y2}) control gates $B_{24}-B_{31}$ which switch the outputs of $\Delta S_2 - \Delta S_5$ lines.

Increments into the $\Delta S_4 - \Delta S_5$ lines which store the data from 40 input channels are entered in the following manner. The increment code pulse (+1 or -1) from the amplifier for comparison of converter voltages (see Chapter VII, Section 1) is stored in flipflop FF_+ or FF_-. Upon application of last pulse P_n of each interval, a code pulse depending on the corresponding flipflop state is transferred through gates B_9-B_{10} to line ΔS_4 or ΔS_5. The ΔS_4 line is designed for storage of the increments from the rapidly varying input channels, which occurs in either of two ten-channel groups in the course of each cycle. For both even and odd cycles, entry into the ΔS_4 line is achieved through gates B_5 and B_6. The ΔS_4 line thus stores the 20 increment codes from 20 channels. The ΔS_5 line

serves for storage of the increments from 20 other slowly varying input channels which are also divided into two 10-channel groups. The entry of both groups into the ΔS_5 line is achieved through gates B_7 and B_8, which are driven to conduct by voltage from the line of flipflop FF_T. This is a control flipflop which is triggered by pulses from the cycle counter (for example, once every 16 cycles). When flipflop FF_T is in the 0 state, the voltage at line p causes gates B_5 and B_6 to conduct; when this flipflop is in the 1 state, the voltage at line q causes gates B_7 and B_8 to conduct for two cycles (one odd and one even); thereafter the flipflop is cleared by a pulse at the 0 input.

For the ΔS_1 current overflow storage line we may also use the circuit of Fig. 206, consisting of two shift registers R_1 and R_2 for the ternary system of increment coding. Each register has 17 bits (20 bits in the y or S register). Since each integrator is shifted by 18 CP pulses (gate B_1 is cut off by pulses P_1 and P_2), the stored code is shifted one bit to the right after 18 CP pulses. At the last pulse P_n, instead of the old overflow code, the new code is written into the registers in the same manner as previously described (see Section 8) by gates B_2 to B_5. The overflow pulses can be selected from the output of gate B_2 or B_3 (through diodes D_1 and D_2) so that the last overflow pulse in the preceding integrator can be transmitted to the input of the next integrator. The overflow pulses called for by the A_x program are selected through gates B_6 and B_7. The selected codes are stored in dynamic flipflops FF_1 and FF_2, which control gates B_8 and B_9. Upon passage of the first pulse P_1 the codes stored therein are transferred to flipflops FF_3 and FF_4 for use as the Δx_+ or Δx_- increments during integration, while flipflops FF_1 and FF_2 are cleared by pulse P_2 for storage of the new codes for the next integrator. The A_x program bits pass through gate B_{10} which is inhibited by pulses P_1 and P_2. The same function is performed by gate B_{11} for the A_{y1} program, which selects the overflow codes for the Δy_+ or Δy_- input of the integrators through gates B_{12} and B_{13}.

Sign reversal of the ΔS overflows coming from the output of any integrator to the overflow line is achieved by a circuit consisting of gates B_{14}-B_{18} and flipflop FF_S. Depending on the code at the bit position of pulse P_1 of the S register, flipflop FF_S is set via gate B_{18} to the 0 state (without sign change) or 1 state (with sign change). Flipflop FF_S controls gates B_{16} and B_{17} to reverse the overflow sign, and at the same time cuts off the passage of the overflow code without sign reversal through gates B_{14} and B_{15}.

The disadvantage of the use of shift registers instead of delay lines is the need of at least one transistor for each bit position of the register. Hence, ten 20-bit registers alone (excluding the control circuits) require a minimum of 200 transistors. The use of standard, inexpensive delay lines or diode-capacitor registers for overflow pulse storage is more practical.

Fig. 207 shows a more detailed diagram of a current overflow storage line and a one-cycle overflow storage line; standard delay line circuits are used (for example, 1-μsec delay at a CP frequency

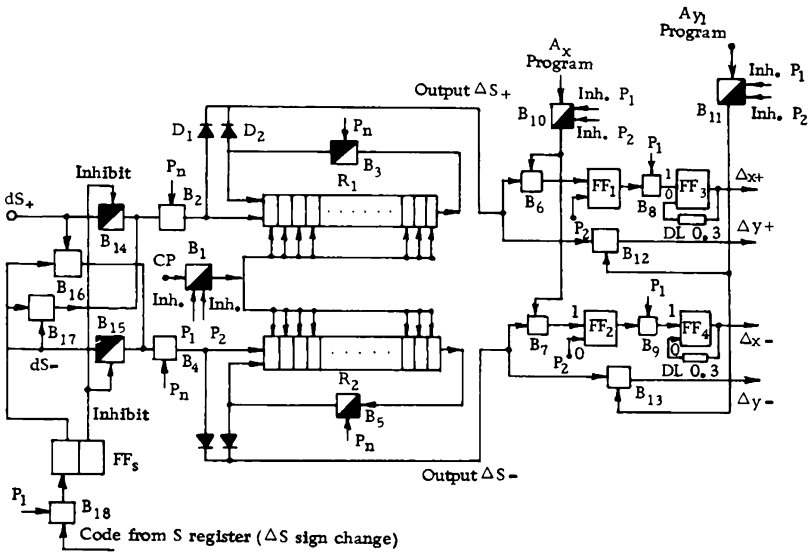


Fig. 206. The use of shift registers as overflow storage lines.

of 1 Mc). Each line contains n-1 elements where n is the total number of bit positions in the y and S registers. The current overflow line normally operates with a precession (i.e., with a 1-bit shift of the stored information after each integrator number), as was discussed above.

The overflow pulse code for the first integrator is located at the right end of the line and all n-1 elements of the line are filled with overflow codes. When the new overflow code of the nth integrator is being entered, the code of the first integrator is erased and is not passed by gate B_1. The overflow code of the n + 1th integrator is written at the location of the overflow code of the second integrator, etc. Thus, the contents of the line are progressively rewritten and the overflows of n-1 integrators can be stored in the line. Since bit pulses P_1 and P_2 are used for control pusposes during readout of the overflow codes, the number of overflow pulses which can be stored in the line is decreased by two codes (i.e., n-3 codes instead of n-1). The codes displaced to the right in the first two elements are not selected during readout. If the DDA is stopped at a given integrator (or a given integration step), it is necessary to preserve the codes stored in the ΔS line, unchanged and without shift for a certain period time. This is achieved by the addition of one more delay element to the ΔS line. The number of elements in the line is thus equal to the number of bits of the y or S register. The code pulses in the line during circulation are not shifted by one bit after each integrator number, but retain their position in the line. The additional DL element is

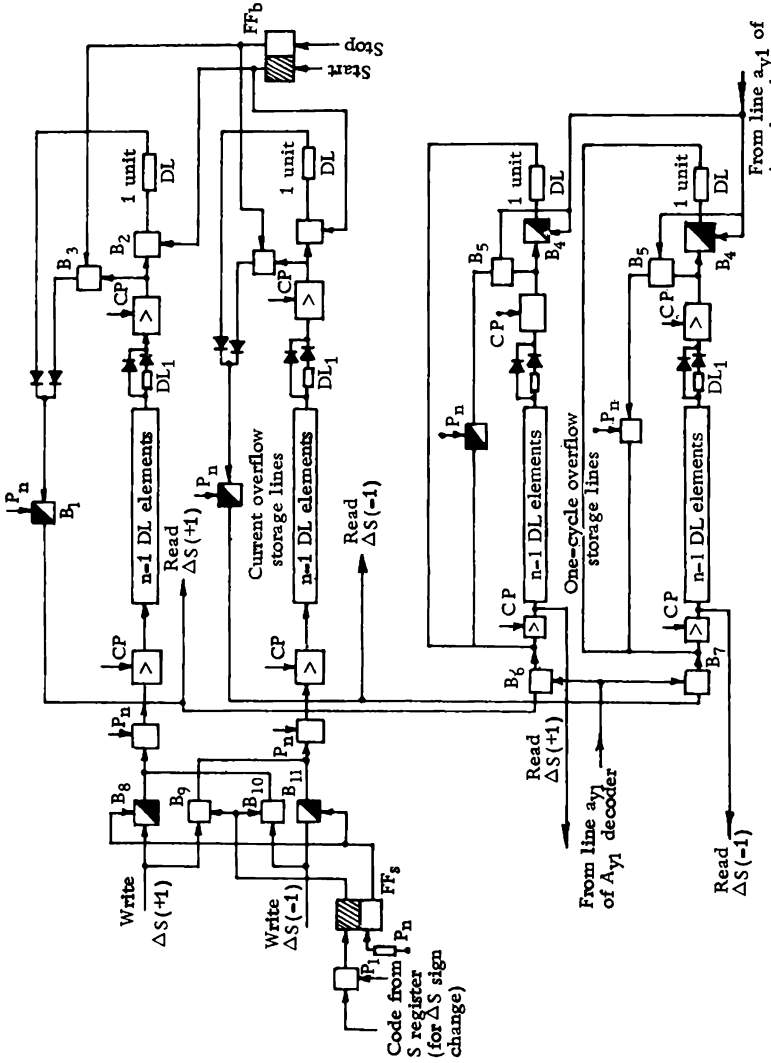


Fig. 207. A more detailed diagram of overflow storage lines.

connected by gate B_2 actuated by flipflop FF_h, which is set to the 1 state on stopping the DDA. At the same time, gate B_3 is cut off and circulation of the pulses occurs over the longer loop.

The principle of "line stretching" is also used in the one-cycle pulse storage line. In this line, the pulse train normally circulates over the longer loop through gate B_4, and the additional delay line element does not displace the train of code pulses. In writing each new code into the line (in the P_n bit position), the information stored in the line is displaced (for this integrator number) one bit to the right due to the circulation of the shorter loop through gate B_5 as gate B_4 is cut off. Gates B_4 and B_5 are controlled by the voltage of a_{y_1} line of the decoder (Fig. 207). The overflow pulse is written into the line at the last P_n pulse through gates B_6 and B_7, which are controlled by the A_{y_1} program. In programming the codes in this line, it must be kept in mind that the 1 bit displacement of the stored information occurs only after the next new code entry.

The signal passing through the line of the n-1 elements is amplified at the end of the line by a transistor amplifier. This amplifier shapes the signal and synchronizes the signal with the CP pulses to correct phase shift which may occur during passage through the line. Delay line DL_1 with a small delay and buffer diodes serves to stretch the pulse at the end of the line. Gates B_8-B_{11} and flipflop FF_S serve to reverse the ΔS sign (as was described for the circuit in Fig. 206).

Instead of shift registers for overflow storage, we may use a simpler circuit (Fig. 208) consisting of individual dynamic flipflops FF, each of which stores the overflow code of one integrator in numerical sequence. Before storing the overflow code of a given integrator, the corresponding flipflop is cleared in the preceding time interval by pulse P_{17}.

The overflow pulses are read out by serial interrogation of the flipflops by timing pulses P_1-P_{18}. The overflow pulse of the last

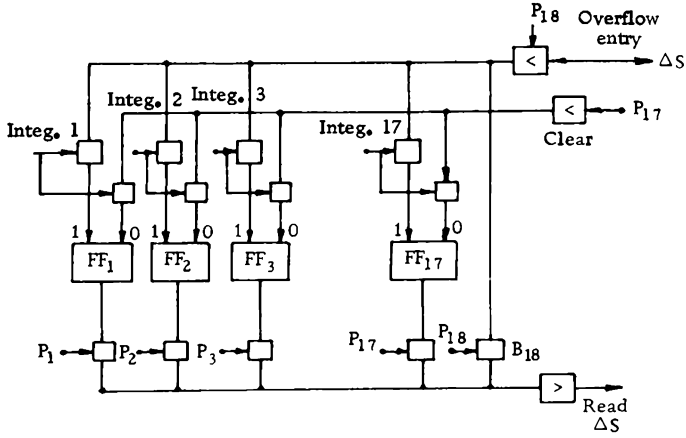


Fig. 208. A ΔS overflow storage line using a group of flipflops.

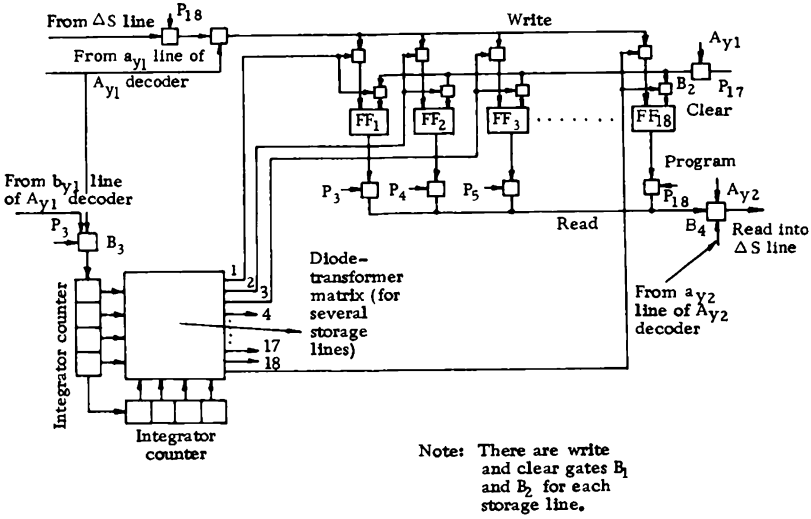


Fig. 209. One-cycle storage lines using groups of flipflops.

integrator is transmitted directly to the overflow read line by gate B_{18} and is simultaneously stored in a dynamic flipflop. This transfers the overflow code pulse from the output of the preceding integrator to the input of the next integrator.

This same principle may be employed for one-cycle storage of overflow codes in a definite sequence by means of dynamic flipflops (Fig. 209). The overflows are entered during the last pulse P_{18} of each integrator through gate B_1 and flipflop input gates; the latter are controlled by the diode-transformer matrix for those integrator numbers whose overflows are to be stored for one-cycle. The diode-transformer matrix is in turn switched by the integrator counter. Each output line of the matrix corresponds to a specific integrator.

During the third time pulse P_3 a pulse is applied to the counter input through gate B_2, which is controlled by lines a_{y1} and b_{y1} of decoder A_{y1} for control of overflow read and write. Each flipflop FF stores the overflow code of a specific integrator as indicated in the A_{y1} program. Before the entry of the overflow code of an integrator, the corresponding flipflop is cleared by time pulse P_{17} through gate B_2. The overflow codes stored in flipflops FF_1-FF_{18} are read out serially via gate B_4 by time pulses P_3-P_{18}. Readout is controlled by the A_{y2} program for the specified integrator and by line a_{y2} of the A_{y2} decoder.

The described read-write scheme for a one-cycle overflow storage does not need a shift register. It is complicated by the

necessity of a diode-transformer matrix with counters for switching of the flipflops. This matrix may be eliminated by insertion of a special program for writing the A_{v3} overflows. However, a single matrix may be used for several one-cycle overflow storage lines if separate write and clear gates B_1 and B_2 are provided for each storage line.

10. STORAGE OF INPUT CHANNEL DATA IN THE MIM

The current values of input quantities arriving from the input converter are stored in the magnetic internal memory (MIM) of the computer. A number of memory locations equal to the number of input channels are allotted to storage of these codes. For example, if there are 40 input channels to be converted to digital form by means of a multi-channel converter, two groups of ten locations each in the y and S matrices of the MIM (Fig. 210). are allocated.

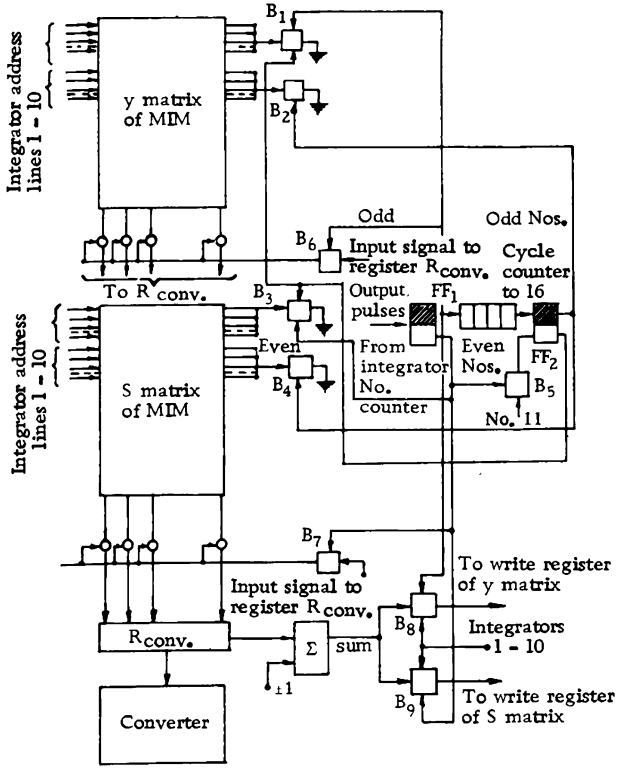


Fig. 210. Read and write operations of input channels.

In each iteration cycle there are 10 time intervals during which the operations of readout, conversion and entry of the new codes for the 10 input channels are achieved. The first ten integrator lines serve for switching of the MIM address lines in which locations the number codes of the ten input channels are stored. These integrator lines may switch the address lines of both groups of each matrix, depending on which of the groups of address lines is connected to the ground. The two groups of address lines (ten lines each) are switched by gates B_1 and B_2 for the y matrix and by gates B_3 and B_4 for the S matrix. Such a method of storing the data for the input channels in two matrices and of switching them in a definite sequence (described below) permits a decrease in the number of required time intervals and number of integrator lines (from 40 to 10). The switching sequence of the four groups of lines is established as follows: the first two groups are switched by gates B_3 and B_4, which are controlled by flipflop FF_1. After each iteration, the pulse from the output of the integrator counter arrives at the common input of flipflop FF_1 and switches it to the other state. When the flipflop is in one state it causes gate B_1 to conduct; in the other state it causes gate B_3 to conduct. Thus, for an odd number in each iteration, gate B_1 conducts and for an even number gate B_3 conducts. Gates B_2 and B_4 switch the two groups of address lines at a frequency, for example, 16 times less than that of gates B_1 and B_3. For this reason, a counter is used; pulses from flipflop FF_1 are applied to the input of this counter. Flipflop FF_2, which is connected after the counter, causes gates B_2 or B_4 to conduct after each 16×2 iterations. After two cycles, during which gates B_2 and B_4 conduct, flipflop FF_2 is returned to the initial state by a pulse from the eleventh integrator line; this pulse passes through gate B_5 which conducts during the second (even) cycle. The locations of address line groups switched by gates B_2 and B_4 are for storage of data from the input channels with relatively slowly varying inputs; conversion of these data is achieved at a relatively low frequency.

The data to be read out of the y and S matrices of the MIM arrive at the converter register (see Chapter VII, Section 1) in accordance with the state of flipflop FF_1. The new number codes obtained from the $\sum$ adder (which adds +1 or -1 to the old number) are entered through gates B_8 or B_9 in the y matrix for odd cycle numbers or in the S matrix for even cycle numbers. The +1 or -1 increments of the input quantities arriving in the computer are stored in the storage lines of the internal memory.

11. DIGITAL INTEGRATING COMPUTER USING ULTRASONIC DELAY LINES

A digital integrating computer may use a magnetic drum as the memory and operate at a pulse-repetition frequency of 100 kc.

The disadvantages of using a magnetic drum in the computer are:

a) The required mechanical precision and balancing of the drum at a high speed of rotation as well as constant spacing between the heads and the drum surface (of the order of 50 microns) regardless of temperature changes.

b) Considerably lower pulse-repetition frequency than in delay lines.

Hence, the speed in a magnetic-drum computer is approximately one tenth the speed obtained from a computer using ultrasonic delay lines. High-speed computers with ultrasonic delay lines may be used to increase the accuracy and volume of computation or to decrease the computing time. An integrating computer using delay lines as an internal storage, in comparison with computers using a magnetic drum storage, may have shorter problem solution times with higher accuracy or the same solution time with more computation.

In comparison with digital arithmetic machines, the advantage in using ultrasonic delay lines in an integrating computer is the continuous, serial selection of number codes from the memory. Here it is not necessary to select the numbers from arbitrary locations of the memory, as is done in machines of arithmetic type. Thus, the memory addressing time and, resultantly, the address selection time in integrating computers are minimal. The use of ultrasonic delay lines in integrating computers permits maximum computation speed with relatively simple computer organization. Solid ultrasonic delay lines of magnesium alloy, quartz lines or magnetostrictive nickel wire may be used in digital integrating computers.

The basic scheme of a digital integrating computer using solid ultrasonic delay lines employs the same principle of serial integration as is used in integrating computers with a magnetic drum. The computer has an internal memory (Fig. 211) consisting of two ultrasonic delay lines 1 and 2 for storage of the number codes of the y and S registers and two electromagnetic delay lines 3 and 4 for increment code storage.

The initial conditions for the integrand y as well as the coded programs for the A_x and A_y addresses are inserted into the internal memory from the control unit of the computer, just as for a computer with a magnetic internal memory (see Chapter V, Section 6). Each of the ultrasonic delay lines contains, for example, 55 registers with 18 bits each. Thus, 990 ($=55 \times 18$) bits are stored in each delay line. If the pulse repetition frequency of the line is 1 Mc, the delay time of each line is 990μsec. Several (5-10) locations in each delay line are allotted for storage of number codes from the converter. A total of 10-20 locations in the two lines is allotted for storage of the number codes from 10-20 converter channels. One location in the ultrasonic delay line is used for storage of the reference pulse from the integrator counter of the temperature control servo system (see Chapter IV, Section 14).

The electromagnetic delay line for overflow storage consists of 17 standard 1-μsec delay lines in series. It serves for storage of the overflow pulse codes at the output of the integrators (ΔS line).

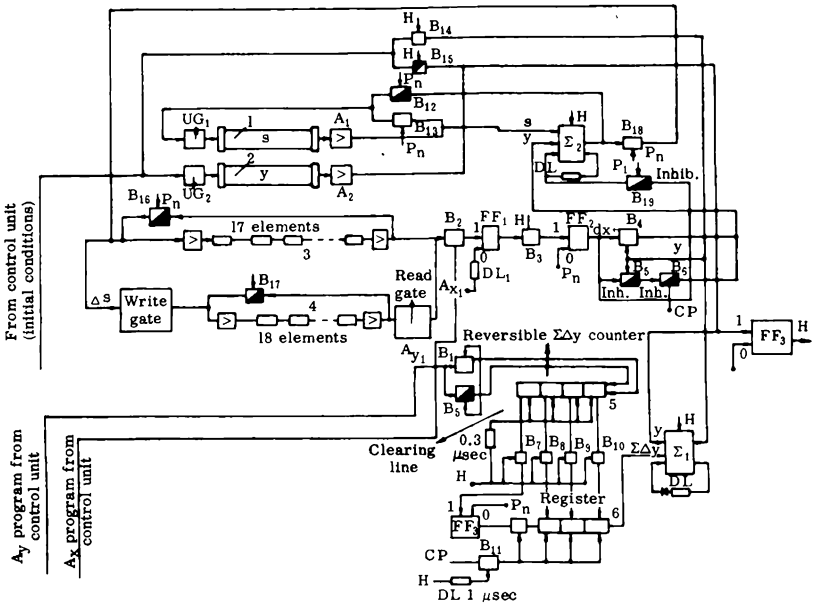


Fig. 211. Integrating unit with ultrasonic delay lines.

At 18-μsec intervals, the overflow pulse arrives at the write gate input, which passes it to lines 3 and 4.

At the input of the ultrasonic delay line there is a special sine-wave generator with a frequency of 10 Mc; this serves to generate ultrasonic oscillations in a quartz plate in contact with the solid delay line. Each of the generators UG_1, UG_2 with a carrier frequency of 10 Mc is modulated bypulses written in the delay line and arriving from the integrating unit.

The clock pulse generator of the computer is at a frequency of 1 Mc. As a CP frequency divider, counters are used which consist of bit counters and integrator counters. Each counter has its diode-transformer matrix (decoder), the output lines of which are directly used for computer control.

Amplifiers A_1 and A_2 at the output of each ultrasonic delay line are used for readout of the pulses stored in the lines. They amplify the voltage oscillations arriving at the amplifier input from the quartz plate at the end of the delay line. These voltage oscillations arise due to the piezo-electric effect in the quartz plate, which converts the ultrasonic oscillations of the quartz into an electrical voltage. After amplification, the modulated carrier of 10 Mc is detected and triangular or rectangular voltage pulses at a repetition frequency of 1 Mc are obtained. These pulses arrive at the integrating unit of the computer through control gates which pass the clock pulses.

The characteristics of a solid ultrasonic delay line of magnesium alloy are listed below.

Characteristics of Solid Delay Lines

Delay time for complete length of line in μsec	500-1000
Pulse repetition frequency in Mc	1
Carrier frequency in Mc (quartz driver frequency)	10
Attenuation in line in db	60-70
Change in delay time with 100°-C temperature change, in μsec	25
Bandwidth in Mc	2

In addition to ultrasonic lines, the integrating unit (Fig. 211) contains the following elements:

a) Line 3 for storing the current ΔS overflows of integrators 3; it consists of 17 electromagnetic delay lines (each with a 1-μsec delay).

b) Line 4 for storing the ΔS overflows of the integrators for 1 integration step; it contains 18 electromagnetic delay lines (each with a 1-μsec delay). The read and write gates in this line ensure selection of the overflow pulses of those integrators which must be written or read in accordance with the program.

c) A single-bit serial adder $\sum_1$ for addition of increments to the ordinates (i.e., for generation of the new ordinate).

d) A single-bit serial adder $\sum_2$ for addition of ordinates (generation of the integral).

e) A counter 5 consisting of 8 stages (including the sign), serving to store the increment pulses of the ordinates (generation of the code $\sum \Delta y$).

f) A 4-bit shift register 6 for serial transfer of the $\sum \Delta y$ number code to the adder.

g) An addressing circuit for selecting overflows from the ΔS line in accordance with the A_x program stored in the computer control unit. The addressing circuit serves to connect the x inputs of the integrators. It consists of two dynamic flipflops FF_1 and FF_2 and gates B_2 and B_3.

h) Addressing circuits for selecting overflows from the ΔS lines in accordance with the A_y program stored in the control unit (a read circuit). It serves to connect the y inputs of the integrators.

i) A circuit with gates B_4, B_5 and B_6 for transfer of the direct or complement code of the number y^* to the $\sum_3$ adder according to the Δx increment code.

j) A dynamic flipflop FF_3 to receive the start pulse.

The operating principle of the above integrating unit using ultrasonic lines is similar to that described in Chapter V, Section 3.

The advantages of solid ultrasonic lines (magnesium, quartz) are the high pulse repetition frequencies and the small number of

electronic components for the internal memory. Their disadvantages are the small, limited memory capacity in each line, high attenuation (particularly for magnesium lines), sensitivity to noise, the necessity for high amplification (due to the low output signal) and the loss of information stored in the lines in the event of brief interruptions in power supply. For these reasons, solid ultrasonic lines (magnesium, quartz) have not found wide application.

12. INITIAL CONDITION REGISTER

Many of the problems to be solved with a digital integrating computer require retention of the initial conditions of the problem until the next computing cycle or accumulation of initial condition increments. This is achieved by the use of an initial condition register (IC) in each integrator. The use of this additional register in an integrator permits repeating at any time (by depressing a button) the solution of a problem with the given initial conditions and inserting new initial data from external units (for example, in the combined operation of an arithmetic machine and an integrating machine). If the initial condition register is combined with an adder as is done in the case of the y and S registers, this permits continuous accumulation of the initial condition increments within the register; thus we obtain therein the new values of initial condition during the problem solution process. This extremely useful operation is often applied in solution of differential equations with given boundary conditions at two points of an unknown curve which is to be found from a family of curves satisfying the given equation (see Chapter II, Section 6).

Figure 212 shows the diagram of an integrator with an initial condition register. The initial condition register IC (for example, a delay line) and the $\sum_{IC}$ adder form a closed loop. The initial condition increments are applied to the second input of the $\sum_{IC}$ adder in the same manner as is done for the y register of the integrator. The presence of the $\sum_{IC}$ adder permits varying the contents of the IC register and accumulating the Δ_{IC} increments therein. Upon the signal to "insert initial conditions," the contents of the initial condition register may be transferred through gate B to the y register, which is previously set to "zero" (or to any desired state). The accumulated initial data are usually inserted into the y register after each complete problem solution (i.e., after a "run" from beginning to end of one of the family of curves satisfying the given equation). A circuit for trapezoidal integration, consisting of y and S registers and three adders $\sum_1$-$\sum_3$, is constructed as described in Chapter I, Section 3. Such an integrator with an initial condition register has a total of three registers (IC, y and S) and four adders.

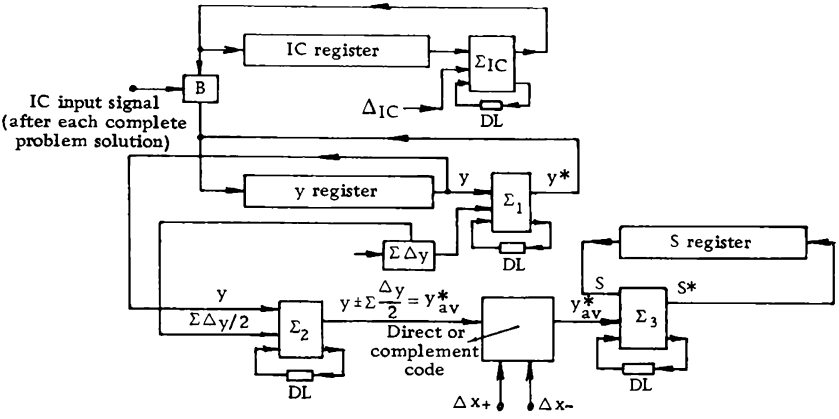


Fig. 212. Block diagram of an integrator with an initial condition register.

13. DOUBLE INTEGRATOR

In the solution of complex problems requiring a large number of integrators it is desirable to perform computing operations simultaneously in two parallel channels by means of so-called double integrators (i.e., to perform the integration operations simultaneously in two integrators). For example, a multiplication can be achieved by the simultaneous operation of two parallel integrators in the time required for one integration. Parallel integrators also permit branching of the computing process into two channels, thereby requiring one-half the time required for a problem solution or, for the same problem solution time, doubling the number of integrators in a computation.

A double integrator (Fig. 213) requires two y and two S working registers; two $\sum\Delta y$ increment counters; four adders (binary coding of increments and rectangular integration); and a common overflow storage line ΔS. For selection of increment codes from the overflow storage line, there are four gates B_1-B_4 which are controlled by the A_{x1} and A_{y1} programs for the first integrator and by the A_{x2} and A_{y2} programs for the second integrator. These four programs permit any overflow code in the ΔS line to be transferred to the input of each integrator. The overflows obtained simultaneously from the outputs of the two integrators are stored in the ΔS line by means of gates B_5 and B_6 controlled by the P_n bit pulse. The overflow code of one integrator is stored in the ΔS line through gate B_6 at the P_n time pulse. The overflow code from the other integrator, which is delayed one bit time by means of flip-flop FF, arrives at the ΔS line in the next P_{n+1} time pulse (through gate B_7). Thus, the number of bits in the y and S code for the double integrators must be increased by one, or must be equal to P_{n+1} bits,

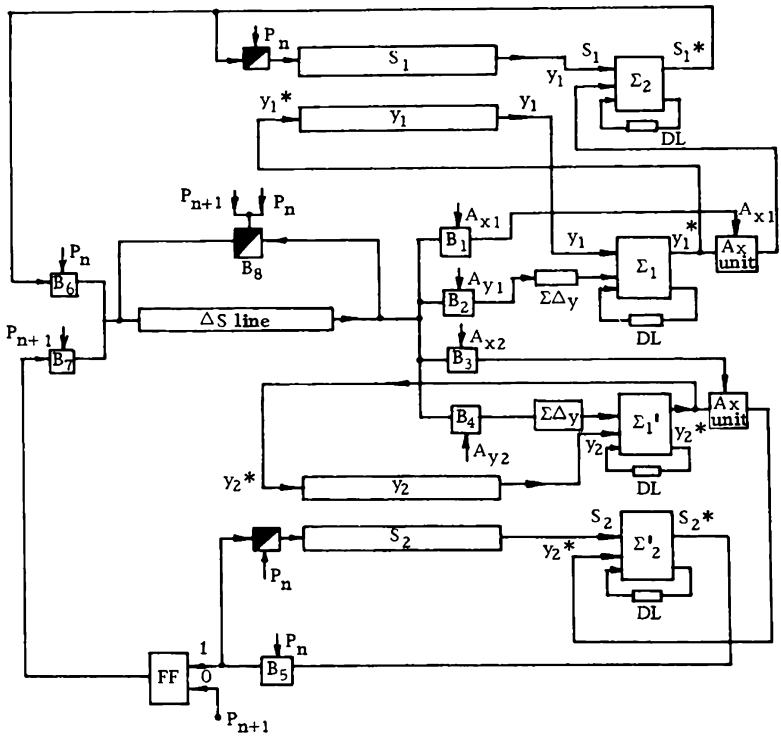


Fig. 213. Block diagram of a double integrator.

with the ΔS line being one bit less (i.e., equal to P_n time intervals). Circulation of the pulses in the ΔS line is interrupted for the P_n and P_{n+1} intervals by gate B_8 which inhibits the reqrite of the ΔS codes. The other circuits of the double integrator is the same as those for a single integrator. A common control unit is used for both integrators. Double integrator may be used for duplication (i.e., solution of the same problem) in order to increase reliability of the computer operation.

Chapter VI

A Serial Decimal Digital Integrating Computer

1. DECIMAL DDA WITH A MAGNETIC DRUM

Scientific and engineering calculations using a DDA are best achieved by communicating with the latter in the decimal number system. In a decimal DDA the integration operation is achieved by means of decimal or binary-coded decimal adders, thereby eliminating internal conversion of decimal numbers to binary numbers and vice versa. The decimal number codes of the y and S registers of an integrator may be written on the tracks of a magnetic drum either in serial-bit parallel-digit form (one binary digit after the other) or in parallel-bit serial-digit form (one decimal digit after the other). In the serial-parallel code for the y and S registers four tracks are used; the parallel four bits on the four tracks are recorded to represent one decimal digit code (e.g., the 5211 code).

In a decimal DDA having a magnetic drum and using series-parallel representation of decimal digit codes (Fig. 214) there are four y-register tracks and four S-register tracks. If the maximum capacity of the y register (and of the S register) is six decimal digits plus one sign digit, then nine digit positions P_1-P_9 are provided for each register; the first two positions (P_1 and P_2) are used for operation control within the computer. Digit positions P_1 and P_2 in the S register are also used for sign reversal of overflow ΔS. Digit positions P_1 and P_2 of the A_x and A_y programs serve for selection of one of the four ΔS overflow storage lines in the same manner as for binary integrators.

If the clock pulses (CP) for reading from and writing on the tracks of the magnetic drum have a frequency of 100 kc, the summation time is 90 μsec. Assume that the track of the magnetic drum is long enough to store 990 code pulses for the 110 integrators plus a dead zone of 10 codes (i.e., a total of 1000 code pulses with a duration of 10 μsec for each pulse). Then the time required for one integration (iteration) for all integrators is 10 μ sec; i.e., the magnetic drum rotates at 6000 rpm or there are 100 iterations per second (with serial-parallel representation of the decimal code). If serial-serial representation is used instead, the number of iterations will be decreased by a factor of 4.

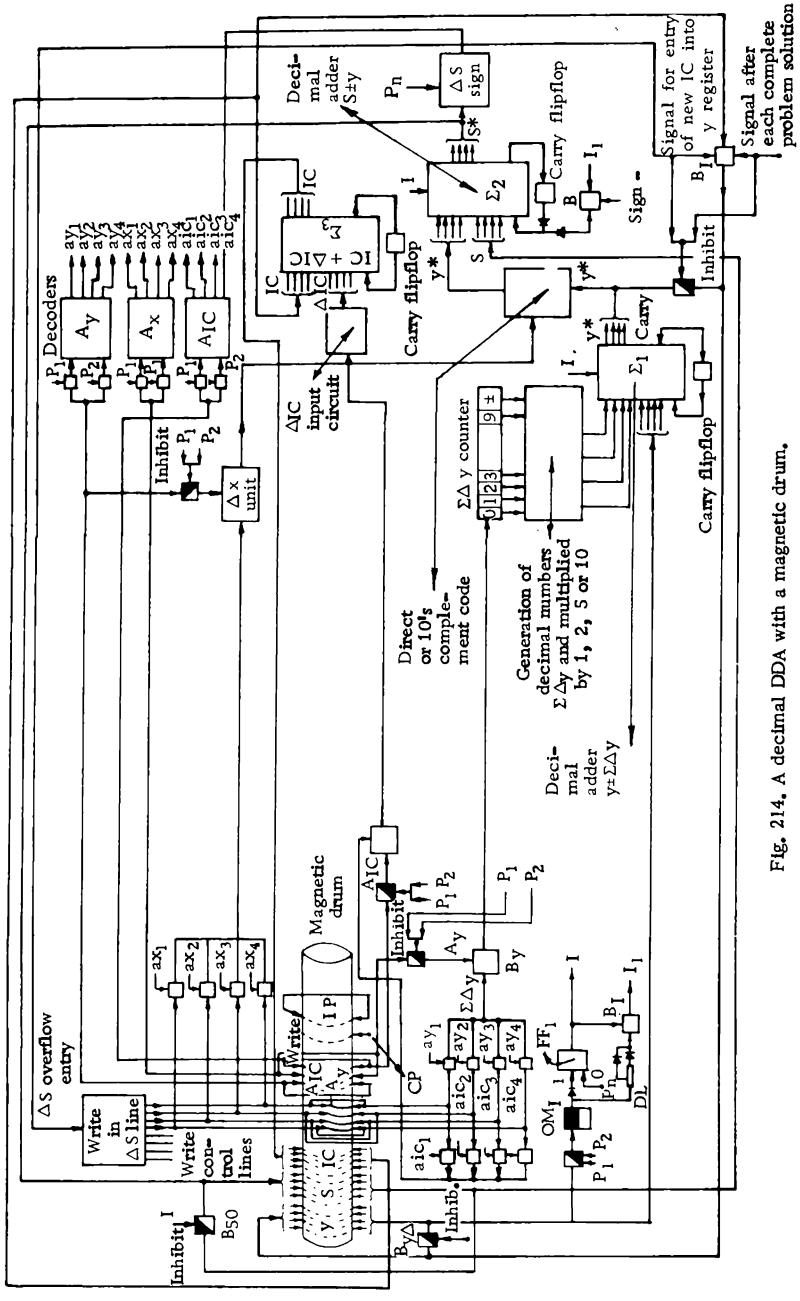


Fig. 214. A decimal DDA with a magnetic drum.

In addition to the y and S registers, there are four more tracks for initial-condition registers (IC registers) for initial-condition storage and for automatic transfer to new initial conditions in the solution of boundary problems (see Chapter II, Section 6). The overflow pulses and increment pulses of the external input channels are stored in the same manner as that previously described (Chapter V, Section 10) for binary coding. Here, each of the short tracks (ΔSline) is one digit shorter than those of y or S registers. If the length of the y register is 9 digits, then the length of each of the ΔS lines is 8 digits; only 6 overflow codes may be stored in each line, since 2 bits are required for switching of the ΔS tracks. Thus, not more than 24 overflow codes may be stored at the same time in the four ΔS lines, which may prove inadequate for DDA operation. An increase in capacity of the ΔS lines can be achieved by using more parallel tracks.

Overflow pulses are read out of the ΔS lines according to the A_x program (for the x inputs), the A_y program (for the y inputs) and the A_{IC} program (for initial-condition registers). These programs are written serially on three tracks of the magnetic drum. Thus, the decimal DDA (Fig. 214) requires a total of 21 tracks, including one track for clock pulses CP and one track for initial pulse IP. In order to increase the capabilities of the DDA several additional tracks are provided for storing several A_x and A_y programs for several problem solutions and several sets of initial values. Due to the fact that the write locations and the read locations are not the same, it is necessary to provide a "floating" address in order to determine their locations. This is achieved by rewriting the initial pulse IP, which is stored on a separate track. At the beginning of a computer run, depression of a button K (Fig. 215) triggers a one-shot multivibrator which, in turn, sets flipflop FF_{IP} to the 1 state, causing gate B to pass the CP pulses to the counter. With the beginning of the CP pulse count in the counter, the initial pulse IP from the one-shot multivibrator is written on its track of the drum. When the drum has almost completed one revolution and has counted all integrator numbers, flipflop FF_{IP} is reset to the zero state by the carry pulse from the integrator number counter and gate B is cut off. Flipflop FF_{IP} causes gate B to conduct again only when the read head reads out the previously written initial pulse IP, which is rewritten in a new location with each revolution of the drum. Since all the write heads are aligned (as are the read heads), the location at which the initial pulse is written determines the start of writing of the codes on all tracks and the counter, regardless of the displacement of information after each drum revolution, correctly determines the locations for all integrators. At the completion of computer operation, the initial pulse IP may be erased by inhibiting gate B_2. The operating principle of the decimal DDA is quite similar to operation of a binary DDA, as has been described (see Chapter V, Section 3). Thus, overflow pulses from the delay lines are selected by the gates controlled by decoder output lines $a_{y1}-a_{y4}$ of the A_y program, where the codes at the first two digits P_1 and P_2 determine the ΔS track

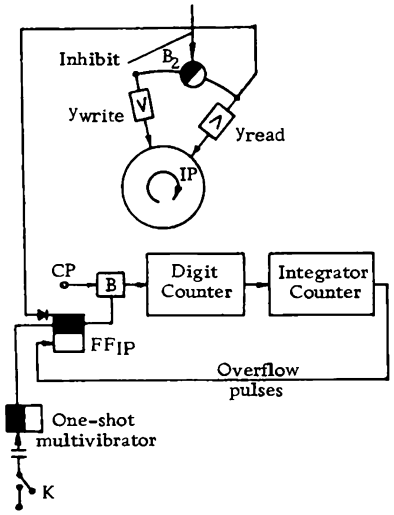


Fig. 215. Diagram of rewrite operation of a magnetic drum.

from which the overflow pulses must be read. If readout must be achieved from two ΔS tracks, an additional A_{y_2} track (not shown in the figure) is needed.

The number of ΔS tracks on the drum and the ΔS addressing circuit depend on whether the overflows are coded binary or ternary. Ternary coding requires twice as many ΔS tracks as binary coding. The overflow codes selected from the ΔS lines through gate B_y (which is cut off for the first two digit pulses P_1 and P_2) are accumulated by the counter to give decimal number $\sum \Delta y$. Depending on the sign of $\sum \Delta y$, this decimal number is converted into a binary-coded decimal number for the decimal adder described below (Section 4). At the same time, it permits multiplication of the number $\sum \Delta y$ by 1, 2, 5 or 10. Thus, it is possible to vary problem solution accuracy and problem solution time by a multiple of 2, 5 or 10. In addition, the capacity of each y and S register may differ by a multiple of 2, 5 or 10.

The decimal code of $\sum \Delta y$ is transferred over four lines to decimal adder $\sum_1$ operating, for example, on the 5211 code, wherein $\sum \Delta y$ is summed with the decimal code of number y. Depending on the time pulse at which the adder is started by initial pulse l, summation of y and $\sum \Delta y$ occurs at the first, second, or other decimal digit of the number code. This permits increasing the magnitude of the increments by a multiple of 10. The position of

initial pulse I in the y number code is decoded by a one-shot multivibrator OM_I and flipflop FF_I. Multivibrator OM_I serves to hold the initial pulse for the duration of one digit (10 μsec); thereafter the trailing edge of the pulse from the one-shot multivibrator triggers flipflop FF_I which generates the I pulses triggering the adder. At the same time, the trailing edge of the pulse from the multivibrator output actuates gate B_1 to allow passage of only the first initial pulse I_1 (delay line DL and the diodes serve to stretch the pulse from the one-shot multivibrator). The initial pulse in the y code may be written in any digit except the first two control pulses P_1 and P_2. The new decimal code y^* from the adder output is written in the y register on the magnetic drum and at the same time is transferred to the input of decimal adder $\sum_2$. Depending on the sign of independent variable increment Δx, a previously described special circuit delivers to the input of adder $\sum_2$ the direct decimal code or the 10's complement of number y^*. The 10's complement is formed by adding +1 to the 9's complement of the number y^*. The addition of +1 may be achieved at the carry input of adder $\sum_2$ by allowing passage through gate B of the first initial pulse I_1 when the value of $\sum \Delta y$ is negative. The resulting new code $S^* = S \pm y$ is written on the tracks of the magnetic drum. The first two digits P_1 and P_2 of the y and S registers are always rewritten without change.

The overflow and increment pulses are written in the ΔS line by a write circuit similar to that previously described (see Chapter V, Section 3). The code in the second digit (P_2) of the S or y register can be used for controlling the write operation. Selection of codes from the ΔS line for Δx input of the integrator is achieved by gates controlled by code lines $a_{y1} - a_{y4}$ of the A_y program decoder (for the first two digit pulses P_1 and P_2). The selected code is stored in the Δx unit, which controls the entry of the direct or 10's complement code of the number y into the decimal adder $\sum_2$.

In addition to the y and S registers (Fig. 214), there is an initial-condition register (IC). This register permits storing the assigned initial conditions of the problem and, in re-running the problem, permits reinsertion of initial conditions into the y register (instead of the existing codes) by depressing a button on the computer control panel. The IC register may serve as buffer storage for connection of the DDA with a digital arithmetic machine for combined operation of DDA and DAC (see Chapter VII, Section 6). It cannot be used for insertion of numbers into the y register from external channels. In the solution of various boundary problems (see Chapter II, Section 6), it is necessary to automatically vary the initial conditions of certain integrators after each complete problem solution (curve "run"). The new initial conditions are obtained by adding to the old conditions certain increments obtained during the problem solution. Selection of increments Δ_I for the

initial-condition registers is achieved from the ΔS overflow storage lines (with one cycle delay) in accordance with a special A_{IC} program on one of the tracks. The first two digit pulses P_1 and P_2 of the A_{IC} program are used for selection of one of the ΔS tracks by the A_{IC} decoder. The operation of writing the overflow pulses from all integrators is achieved by previously described circuits (Chapter V, Section 3). During each iteration, increments are stored in the IC registers of the integrators assigned in the IC program. New initial values are thus obtained and then entered into the y registers of the corresponding integrators after each complete problem solution (i.e., after each "run" for one curve of the family). The new IC codes read from the IC tracks are entered into the y register through gate B_I for those integrators in which they are to be written. Therefore, gate B_I is controlled by one of the output lines of the decoder for the A_{IC} program which specifies the required integrators. In writing new IC codes, gate B_I^1 inhibits writing of the number codes of the y register. Gates B_I and B_I^1 are also controlled by a signal from the computer after each complete problem solution, which enables the operation of writing new codes in the y register. This signal may be obtained, for example, at the moment when the argument in the y register reaches a preassigned value or when the carry from an iteration counter corresponds to a definite number of integration steps.

The Δ_I increments and the number code in the y register are summed by decimal adder $\sum_3$. The circuit for entering Δ_{IC} into adder $\sum_3$ is the simplest when Δ_{IC} can be only +1 or -1. In this case, +1 is added directly to the least significant decimal digit and -1 is added by adding the 10's complement of one.

The above described decimal DDA may also employ a faster memory (for example, relatively inexpensive magnetostrictive delay lines with a pulse repetition frequency of 0.5 Mc). In using delay lines, each track on the magnetic drum is replaced by a separate line. More stringent requirements are then placed on the operation of the decimal adder and other elements. The computing rate of the decimal DDA may thus be increased by a factor of five or larger.

2. 5211-CODE DECIMAL ADDER

A one-digit decimal adder designed for serial bit-by-bit addition and subtraction of coded decimal numbers is now described. The chosen code for the decimal adder is the four-bit 5211 code, which is self-complementing. A self-complementing decimal-digit code is one that, if the 1's and 0's of each coded digit are replaced respectively by 0's and 1's, the resulting digit is the 9's complement of the given coded digit. The use of a self-complementing code considerably simplifies the subtraction operation because, as in the binary number system, subtraction reduces to adding the 9's

complement of the subtrahend to the minuend and adding one to the least significant digit of the minuend. In addition, the 5211 code, as will be seen below, permits the use of a relatively simple adder.

The positive and negative decimal digits in the 5211 code are listed in Table 10.

TABLE 10

Decimal codes

Decimal digits	Sign code	Direct code				Complement code				Decimal digits	Sign code	Complement code			
		Digits													
		5	2	1	1	5	2	1	1			5	2	1	1
0	[0]	0	0	0	0	1	1	1	1	−1	[1]	1	1	1	1
+1	[0]	0	0	1	0	1	1	0	1	−2	[1]	1	1	1	0
+2	[0]	0	1	0	0	1	0	1	1	−3	[1]	1	1	0	0
+3	[0]	0	1	1	0	1	0	0	1	−4	[1]	1	0	1	0
+4	[0]	0	1	1	1	1	0	0	0	−5	[1]	1	0	0	0
+5	[0]	1	0	0	0	0	1	1	1	−6	[1]	0	1	1	1
+6	[0]	1	0	1	0	0	1	0	1	−7	[1]	0	1	1	0
+7	[0]	1	1	0	0	0	0	1	1	−8	[1]	0	1	0	0
+8	[0]	1	1	1	0	0	0	0	1	−9	[1]	0	0	1	0
+9	[0]	1	1	1	1	0	0	0	0						

The decimal adder (Fig. 216) consists of three one-bit full adders $\sum_1$, $\sum_2$ and $\sum_5$ (three inputs), one half adder $\sum_1'$ (two inputs) and coincidence gates B_1–B_{12}. The 5211 decimal code is applied to inputs x, y, x_1, y_1, x_2, y_2, and x_5, y_5 of the four binary adders. Carries W_1' and W_1 from the outputs of adders $\sum_1'$ and $\sum_1$ are applied to the W_1 input of adder $\sum_2$. In addition, carry to adder $\sum_2$ may occur through gate B_{12} if code 1 is simultaneously obtained at output lines C_1 and C_2 of adders $\sum_1'$ and $\sum_1$. The sum and carry outputs of adders $\sum_1'$–$\sum_5$ control gates B_1–B_{11} by which the sum is generated in correct decimal code.

For example, in the presence of signals at lines C_1 or C_1', gate B_1 passes a signal giving a decimal value of 1 if there are no

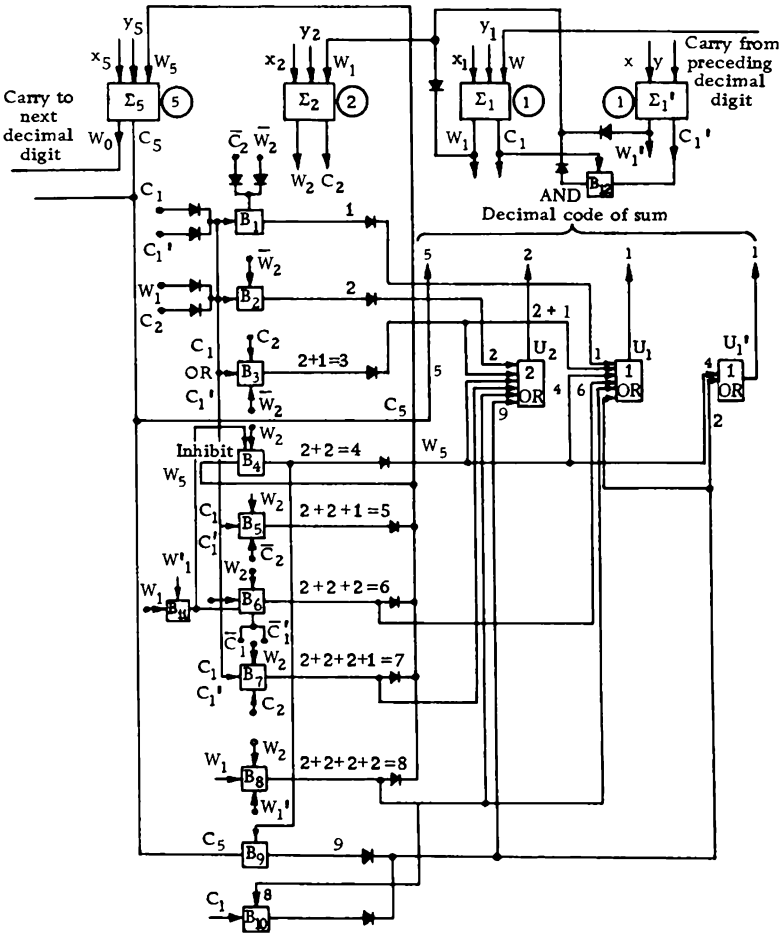


Fig. 216. 5211-code decimal adder.

signals on lines C_2 and W_2 from adder $\sum_2$ ($\bar{C}_2$ and $\bar{W}_2$ in the diagram denote not C_2 and not W_2).

Gate B_2 gives a decimal value of 2 if there are signals on lines W_1 or C_2 and if there is no signal on line W_2 from adder $\sum_2$. Gate B_3 gives a decimal value of 3, (2 + 1), if there are signals on lines C' or C_1' and simultaneously on line C_2 but not on line W_2. Gate B_4 gives a decimal value of 4, (2 + 2), if there is a signal on line W_2 and signals on lines W_1 and W_1' (via gate B_{11}) and if there is no carry W_5 to adder $\sum_5$.

Gate B_5 gives a value of 5, (2 + 2 + 1), if there is a signal on line W_2 and simultaneously on line C_1 (or C_1') but no signal on

line C_2. Gate B_6 gives a value of 6, (2 + 2 + 2), if there is a signal on line W_2 and on line C_2 (or coincidence of signals on lines W_1' and W_1 in gate B_{11}) but no signals on lines C_1 or C_1'. The decimal values of 7, (2 + 2 + 2 + 1), and 8, (2 + 2 + 2 + 2), are similarly formed by gates B_7 and B_8. Gate B_9 gives a decimal value of 9 if there is coincidence of signals on line C_5 and at the output of gate B_{10} (4 + 5 = 9). This value 9 also occurs at gate B_{10} if there are signals on line C_1 and at the output of gate B_8 (1 + 8 = 9). The decimal code proper of the sum is obtained from the outputs of the OR circuits (U_1, U_1', U_2) and from line C_5 of adder $\sum_5$. The inputs of these OR circuits are connected to those outputs of gates B_1 and B_{10} which give the sum in 5211 code.

A few examples of addition of decimal digits in the 5211 code are shown below.

Example 1

	Carry			
	1			
+9	Code 1	1	1	1
+4	Code 0	1	1	1
+13		0	1	1	1

Carry to next significant digit is 1

The addition gives the sum code 0110 corresponding to the decimal digit 3 (= 2 + 1) and a carry from this digit.

Example 2

	Carry			
	1	1		
+8	Code 1	1	1	0
-3	Code 1	0	1	1
+5		1	0	0	0

Carry to next significant digit is 1

Addition of a negative number is equivalent (as in the binary system) to adding the 10's complement of the number. The 10's complement of the number is obtained by replacing 1 with 0 and 0 with 1 in the direct code (this gives the 9's complement) and adding one to the least significant digit of the 9's complement.

In adding or subtracting numbers having several decimal digits, the carry to the next more significant digit is applied to the W input of the next adder.

The number code is preceded by a sign digit (0 for positive and 1 for negative). The carry from the most significant digit is added to the sign digit, as in the case of binary addition or subtraction.

We note that in these examples the sum is not the sum of binary addition, but is the sum of ones, twos and fives (the summed numbers) obtained by gates B_1-B_{10} of the above described decimal adder. Thus, for example, in adding the digit codes 0111 (four)

and 0011 (two), the sum is six; in adding the codes 0100 and 0011, the sum is four, etc.

The decimal digit in 5211 code can be simply converted to the ordinary decimal digit (1 to 9) by means of the gating circuit shown in Fig. 217. In this circuit, only one of the gates B_1–B_9 conducts and this gate gives the decimal value from 1 to 9 which corresponds to the given coded decimal digit.

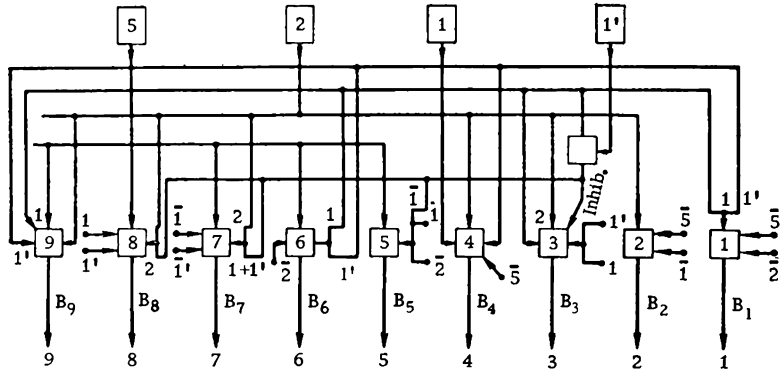


Fig. 217. Decimal digit code converter.

For example, gate B_1 conducts only when there is a code 1 in either of the first two least significant bits of the 5211 code and is cut off if there is a code 1 in the fives or twos bit.

Conversion of a decimal digit to the 5211 code (for entering initial data into the computer) is achieved in a similar manner; there are four gates which are caused to conduct by the decimal value according to the value of the gates in (5, 2, or 1 as listed in Table 10). Thus, the 5211 code gives a simple conversion of input and output decimal data. The sign digit of the decimal adder in a digital differential analyzer uses code 1 and code 0 to represent the positive and negative numbers respectively (i.e., the complement of the sign code in Table 10). In this case, overflow in code 1 denotes a positive increment (+1) and overflow in code 0 a negative increment (-1). In ternary coding, the rules for obtaining the increment codes for decimal numbers remain the same as those for binary numbers.

3. EXCESS-3-CODE DECIMAL ADDER

The excess-3 code in representing decimal digits is obtained by adding the binary number 3 to the decimal digit expressed in the usual 8421 binary code. The excess-3 code for the 10 decimal digits is shown in Table 11.

Table 11

Excess-3 Decimal Codes

Decimal digits	Excess-3 code	Decimal digits	Excess-3 code
0	0011	5	1000
1	0100	6	1001
2	0101	7	1010
3	0110	8	1011
4	0111	9	1100

Although this code is not a weighted code (i.e., the sum of the binary digits therein is not equal to the decimal digit), it is a self-complementing code. For example, the 9's complement of the decimal digit 4 (0111) is 5 (1000); the 9's complement of the decimal digit 3 (0110) is 6 (1001).

The rules for addition and subtraction of decimal digits in excess-3 code are simple. The decimal carry to the next decimal digit is obtained automatically from the most significant bit in binary addition of the codes. Since the code of addend or augend digits has an excess of 3, the resulting binary code of the sum has an excess of 6 and a correction of minus 3 must be applied. Let us examine two cases of addition of two decimal digits in excess-3 code.

a) First case. The sum of two decimal digits is less than 10. In this case, the code of the sum in binary addition is in excess-6 code and is incorrect. In order to obtain the correct excess-3 code, it is necessary to subtract 3 from the incorrect sum. This may be achieved by adding the binary number 13 and subtracting 16 (i.e., ignoring the carry from the fourth most significant bit).

Example 1.

		1 carry	
+4	Code	0111	
+3	Code	0110	
+7 No carry		1101	(digit 7 in excess-6 code)
	+		
		1101	(add binary number 13)
Carry 1 here to next decimal digit is discarded		1010	(digit 7 in excess-3 code)

b) Second case. The sum of two decimal digits is greater than or equal to 10. In this case there should be a carry to the next decimal digit, which in binary addition is obtained from the fourth most significant bit. Since this carry corresponds to the number 16 instead of 10, the sum code does not have an excess 6 and is incorrect. In order to obtain the correct excess-3 code, binary number 3 (i.e., 0011) is added to give a correct sum code.

Example 2.	Carry 11	
+8	Code 1011	
+6	Code 1001	
+14	0100	
	+ 0011	
Carry 1 to the next decimal digit is correct	0111	Add binary number 3 (digit 4 in excess-3 code)

Thus, the rules for addition of decimal digits in excess-3 code are as follows. It must first be established whether or not there is a carry from the most significant bit of a decimal digit to a more significant digit. If there is a carry, binary number 3 is added to the uncorrected sum and the carry is added to the next more significant decimal digit. If there is no carry, binary number 13 is added to the uncorrected sum and the resulting carry is ignored.

Subtraction of decimal numbers in excess-3 code is performed in the same manner as the subtraction of binary numbers. The subtrahend is converted into 10's complement code. The carry operation and the addition of binary number +3 or +13 remain the same as in the case of addition.

Example 3.

+425		425
-229		770
+196	or complement	+1195
		1
		1196

In excess-3 code:

Carry	Carry	No Carry	No Carry
1	11		
[0]	0111	0101	1000
[1]	1010	1010	0011
[0]	0001	1111	1011
0	0011	1101	1101
[0]	0100	1100	1000
			+1
Code of [0] 196	0100 (1)	1100 (9)	1001 (6)

The block diagram of a decimal adder in excess-3 code is shown in Fig. 218. Addition of two excess-3 coded digits A and B (four bits each) is achieved by four half adders $\sum_1 - \sum_4$ and then by four half adders $\sum_1' - \sum_4'$ with due consideration of bit carries and digit carry.

Each pair of half adders $\sum_1$ and $\sum'_1$ and the like forms a single-bit full adder which could be replaced by a conventional full adder if a special carry circuit to the next higher decimal digit were not employed. The special carry circuit is a five-input AND-coincidence circuit to which are applied the sum outputs of half adders $\sum_1 - \sum_4$ and the carry W from the previous decimal digit. Either the output of the coincidence circuit or the carry output from the adder $\sum_4$ is the digit carry. This arrangement reduces propagation time of the decimal digit carry.

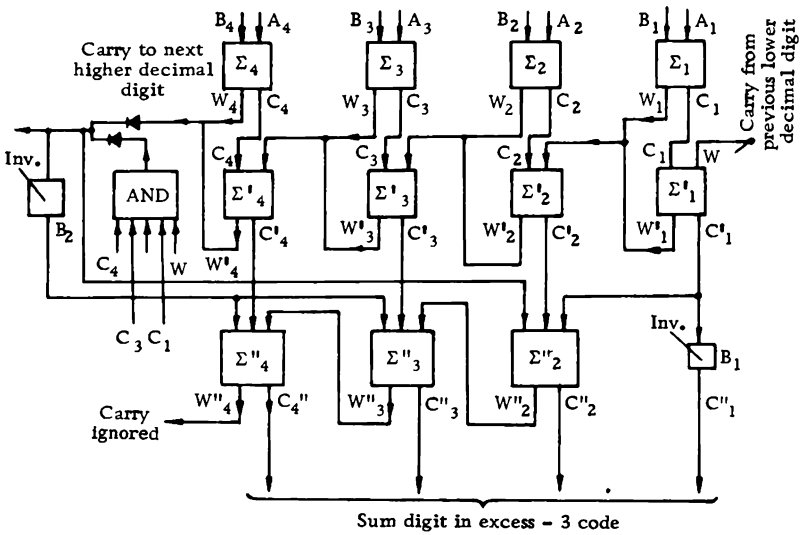


Fig. 218. Decimal adder in excess-3 code.

The sum lines C'_1-C'_4 of half adders $\sum_1$-$\sum_4$ in Fig. 218 give an uncorrected sum. Depending on whether there is a digit carry or not (from line *a* in the figure), a correction of 3 or 13 must be added to the uncorrected sum. This correction is achieved by means of three full adders $\sum''_2 - \sum''_4$ and inverters B_1 and B_2. Since binary numbers 3 and 13 are 0011 and 1101, a 1 is always added to the lowest bit position. This add-1 operation is achieved by complementing the C'_1 output. Furthermore, if there is a digit carry, 1 is applied to one of the inputs of adder $\sum''_2$. If there is no digit carry, 1 is applied to the inputs of adders $\sum''_3$ and $\sum''_4$. This 1 is obtained from the output of inverter B_2. The carry output of adder $\sum''_4$ is ignored.

The above described decimal adder using excess-3 code requires eight half adders, three full adders, two inverters and one five-input AND-coincidence circuit for fast carry. It thus requires more electronic components than a 5211-code decimal adder (see Chapter VI, Section 2).

The disadvantages of excess-3 code are now summarized: (a) representation of the digit zero is not zero in all four bits of the code, which complicates machine design and implementation of certain arithmetic and logic operations in the computer; (b) the excess-3 code is not a "weighted" code (i.e., the sum of the four binary digits is not equal to the decimal digit), which causes further complication; (c) for converting a decimal code to the excess-3 code and vice versa special decoders are required with a considerable number of diodes.

4. INCREMENT SUM $\Sigma_\Delta y$ FOR A DECIMAL ADDER

The algebraic sum of the increments $\sum \Delta y$ must be represented in the decimal code adopted for the decimal adder. For flexibility in scaling the contents in the y register, it is also necessary to provide for multiplication of the increment sum $\sum \Delta y$ by a factor of 1, 2, 5 or 10. The register capacity is thus

$$a \cdot 10^{n-1},$$

where a is the factor (1, 2, 5 or 10) and n is the number of decimal digits in the y register. Let us assume, for example, that the maximum number of inputs for the integrator is 7 and that the increment sum $\sum \Delta y$ ranges from 0 to ± 7 units. In this case, we have the following decimal values (the direct code for positive numbers and 10's complement code for negative numbers) shown in Table 12.

It is seen from Table 12 that upon multiplication of the positive $\sum \Delta y$ by 2 the value of the leftmost decimal digit is 0 or 1, while upon multiplication by five the value is 0, 1, 2, 3 or 4. For negative $\sum \Delta y$, upon multiplication by 2 the value of the leftmost decimal digit is 8 or 9 and upon multiplication by 5 the value is 5, 6, 7, 8 or 9. Upon multiplication by a given factor (1, 2 or 5), the values of the rightmost decimal digits are the same for the corresponding positive and negative sums $\sum \Delta y$.

Figure 219 shows the diagram in which the codes for positive and negative $\sum \Delta y$ in Table 12 are used. Count of code pulses Δy selected from the ΔS line at the integrator input is achieved by

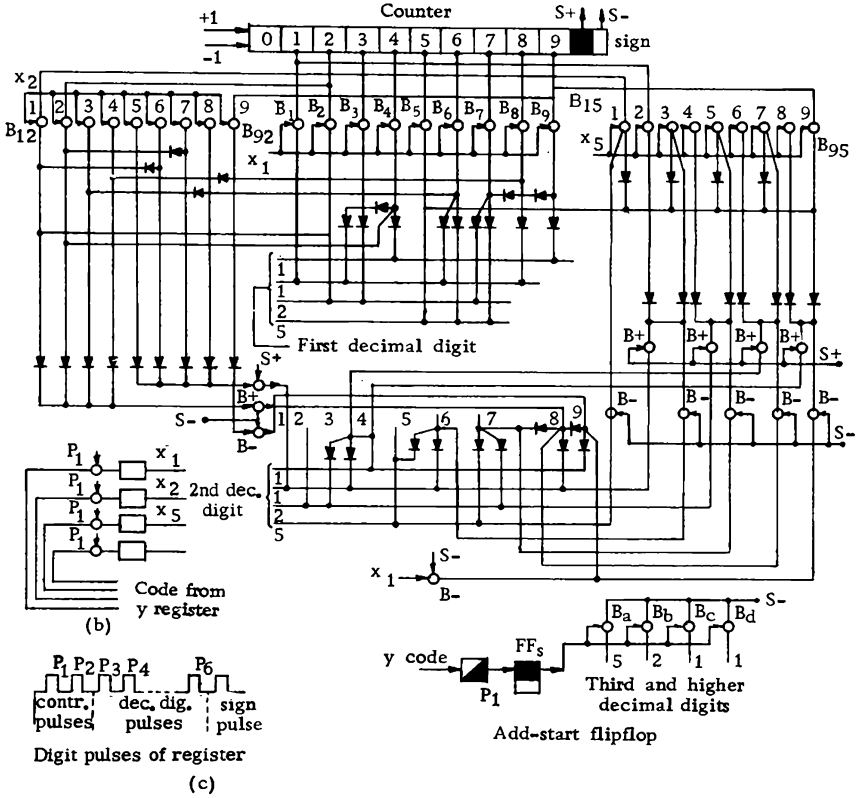


Fig. 219. Generation of increment sum for a decimal adder.

a reversible decimal-digit ring counter (see Chapter III, Section 14) having ten stages plus a sign stage. In this ring counter, positive values of $\sum \Delta y$ are represented by direct code and negative values by complement code.

Gates B_1, $B_2 \ldots B_9$ are for multiplication by 1, gates B_{12}, $B_{22} \ldots B_{92}$ for multiplication by 2 and gates B_{15}, $B_{25} \ldots B_{95}$ for multiplication by 5. The left and right decimal digits are coded in 5211 code and transferred to the adder. The values of these digits are those listed in Table 12 for a decimal adder. Each group of gates conducts upon arrival of the code pulse written in the first (P_1) decimal position of the y register. The code pulse is stored in one of four dynamic flipflops FF_1-FF_4 (Fig. 219b) which directly control the gates.

The state of one of flipflops FF_1-FF_4 corresponds to a factor by which $\sum \Delta y$ is multiplied. Depending on the location of code 1 written in the first decimal position of the y register, a flipflop corresponding to one of the factors 1, 2 and 5 is triggered. For

TABLE 12
Positive and Negative Values of Increment Sums

Contents of the counter	Positive values									Negative values								
	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	7	8	9
$\sum \Delta y$	+1	+2	+3	+4	+5	+6	+7	+8	+9	-9	-8	-7	-6	-5	-4	-3	-2	-1
Decimal code for $\sum \Delta y \cdot 1$	1	2	3	4	5	6	7	8	9	91	92	93	94	95	96	79	98	99
Decimal code for $\sum \Delta y \cdot 2$	02	04	06	08	10	12	14	16	18	82	84	86	88	90	92	94	96	98
Decimal code for $\sum \Delta y \cdot 5$	05	10	15	20	25	30	35	40	45	55	60	65	70	75	80	85	90	95

multiplication by 10, the first decimal digit of code $\sum \Delta y$ is summed with the second digit of the y register code. To achieve this, adder $y \pm \sum \Delta y$ is actuated by a flipflop which is triggered by pulse P_3 written in the first decimal digit of the register y. Since the code of sum $\sum \Delta y$ in the second decimal digit depends not only on the multiplication factor but also on the sign of $\sum \Delta y$, the sign digit of the counter has two control lines S_+ and S_- representing, respectively, code 0 (positive numbers) and code 1 (negative numbers). These lines control gates B_+ or B_-, which pass to the second decimal digit $\sum \Delta y$ the code listed in Table 12 for multiplication factor 1, 2 or 5. For the third decimal digit and all more significant digits, gates B_a-B_d set code 1 in all these digits of the adder (on 5211 code); this gives a value corresponding to the digit 9 in 5211 code. These gates are driven by the add-start flipflop FF_s.

For each of the integrators, the circuit in Fig. 219 permits selection of one of the multiplication factors 1, 2, 5 and 10 by appropriate coding of the control pulses in the y register.

Chapter VII

Multichannel Analog-to-Digital Converters

In a digital integrating computer continuous variables must be converted to digital code or to increments of measured quantities by analog-to-digital converters or digitizers. The known methods are mostly for converting voltage amplitude or angular shaft position to digital code. There are four types of voltage digitizers:

a) Pulse-interval voltage converter. In this converter a given voltage is compared with a linearly varying sawtooth voltage. At the moment of coincidence of the compared voltages a pulse is generated to establish a time interval from the beginning of the linear sweep of the sawtooth voltage (initial pulse) to the moment of coincidence. This time interval is proportional to the amplitude of the converted voltage and is converted to digital code by means of a counter which counts pulses of a reference frequency for the given time interval. This method of conversion has found wide application due to its relative simplicity and high accuracy. A disadvantage is the need of counting pulses during each conversion; this counting limits the operating speed.

b) Converters based on comparison of a voltage with reference voltages. In this method the voltage under conversion is compared successively with a number of reference voltages. The conversion accuracy depends on the accuracy of the reference voltages and the accuracy of comparison.

c) Direct voltage-to-digital converter. In this converter the beam of a cathode-ray tube is deflected by a voltage applied to its deflecting plates. A code "mask" on the screen of the tube converts the deflected beam to a digital code.

d) Voltage converters employing feedback principle.

In this chapter only a few multichannel converters using the principles of feedback and increment accumulation are described. For each voltage increment there is a corresponding increment pulse which is sent to the digital integrating computer. This method of conversion in the form of increments permits a considerable increase in conversion speeds and is thus important in operating with a large number of channels.

1. A CONVERTER USING DIODE SWITCHES

A multichannel electronic converter converts the voltages obtained from various transducers and measuring devices to a digital form. Ordinary potentiometers with a d-c voltage supply can serve as the transducers. The input values are inserted into the computer in the form of increment pulses at a rate of 500-1000 per second for each input channel. Switching of input channels is achieved by an electronic circuit which, for example, switches through all channels every 1-2 msec.

The number code of each input is stored in the computer memory and is replaced with a new code as the input changes. The change in input is inserted into the computer in an incremental form; it may also be represented in the form of a complete number having, for example, 10-12 bits.

The multichannel electronic converter consists of the following principal units: the converter proper; null amplifier; shift register; adder; circuits for converter control and data entry into the computer. The converter proper, shown in Fig. 220, converts the number code to a proportional voltage. The number code is stored by a ten-bit electronic pulse counter or a code register. Each bit position of this register controls an electronic gate. The gates are represented in the diagram as switches $K_1, K_2, K_3 \ldots K_{10}$. The code register consists of a number of dynamic flipflops which, with code 1 in a given bit position, deliver a voltage cutting off the gates (i.e., opening the switches). Thus, with code 1 in all bit positions of the register, switches K_1-K_{10} are open (in the off state) and with code 0 in all bit positions they are closed. For any other number code, only those switches whose corresponding bit positions store code 1 are open and the remaining switches are closed.

Switches K_1-K_{10} pass currents which are proportional to the "weight" of the given code bit. For this purpose the value of the resistor in each bit position is inversely proportional to the weight of the binary code of the given bit. If the current of the first (least significant) bit is i, the current of the second bit is $2i$, the third $4i$, the fourth $8i$, and the tenth $512i$.

The value of the resistor for the first bit is $512R$, for the second bit $256R$ and for the tenth bit R.

The sum of the currents flowing through resistors R, $2R$, $4R\ldots$ etc., with switches K_1-K_{10} all closed, passes through a common resistor with a value R_0. A current flows in resistor R_0 as a result of voltage U_n, which is taken from the potentiometer of the input channel transducer.

The value of resistor R_n (in series with voltage U_n) is so chosen that with maximum value of U_n the current flowing through R_n will supplement the current flowing through resistor R_0, resulting in a maximum value corresponding to the current through all closed switches K_1-K_{10}. With voltage $U_n = 0$ all bit positions of the register contain code 0 and all switches K_1-K_{10} are closed. The currents flowing through resistors R, $2R$, $4R\ldots$etc. produce at resistor R_0 a

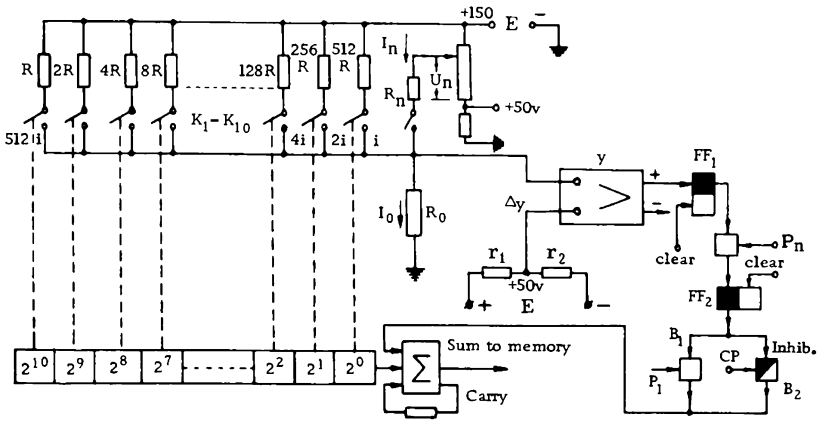


Fig. 220. Conversion of number code to voltage.

voltage drop U^* which remains constant during operation of the circuit (e.g.,+50 v). When voltage U_n is maximum, all bit positions of the register contain code 1 and switches K_1-K_{10} are open. In this case, the current through resistors R_n and R_0 is equal to the sum of the currents which earlier flowed through all resistors R, $2R$, $4R$... etc. This current produces at the resistor R_0 a voltage drop equal to U^*. Thus, the sum of currents I_Σ flowing through resistors R, $2R$, $4R$... etc. and the current I_n flowing through resistor R_n under the influence of potentiometer voltage U_n remains constant or

$$I_\Sigma + I_n = \text{const.} \tag{1}$$

If one of the currents changes value, the other undergoes a compensating change, maintaining a constant result. Since the voltage drop at resistor R_0 in the equilibrium state remains constant and equal to U^*, it is evident that the currents through resistors R, $2R$, $4R$... etc. will be:

$$\left.\begin{aligned} i_1 &= \frac{E-U^*}{R}; \quad i_2 = \frac{E-U^*}{2R}; \\ &\cdots\cdots\cdots \\ i_n &= \frac{E-U^*}{2^{n-1}\cdot R}. \end{aligned}\right\} \tag{2}$$

Hence it is seen that the resistors in the circuit at each bit position of the converter must be chosen according to the weight of the given bit position of the code.

A differential amplifier is used to compare the voltage drop at resistor R_0 with the fixed voltage from the resistor divider

R_1 and R_2 (equal to +50 v). The voltage taken from the potentiometer varies, for example, from +50 to +150 v.* The +50-v potential of the potentiometer corresponds to $U_n = 0$ and the +150-v potential corresponds to $U_{n\,max}$. With a change in voltage U_n, current I_n flowing through resistor R_n also varies and the voltage drop at resistor R_0 departs from its reference value of +50 v, at which the differential amplifier is balanced.

At the amplifier input there arises an error voltage ΔU which is amplified and, depending on the error sign, causes gate B_1 or B_2 to conduct, admitting pulses to the adder and register until the error voltage ΔU is zero. With each voltage comparison step, a unit pulse, depending on the ΔU error sign, is added to or subtracted from a number code stored in the register by means of the adder. This pulse arrives at the adder input through gate B_1 if 1 is to be added or through gate B_2 if 1 is to be subtracted. Subtraction of 1 is equivalent to adding code 1 to all bit positions of the number code in the register and is achieved by gate B_2, which conducts only when the amplifier output is a negative voltage. In this case, flipflops FF_1 and FF_2 are in the 0 state and there is no inhibit pulse present at gate B_2 (from flipflop FF_2). During the summing operation the CP pulses that pass through gate B_2 are added with the number stored in the register and the sum (from the adder output) is stored in the computer memory.

The change in number code in the register by addition or subtraction of one pulse occurs until the error voltage at the amplifier input is zero. The code stored in the register then corresponds precisely to the potentiometer voltage U_n. Figure 221 shows a circuit diagram for one bit position of the converter. The bit-position resistor R, through which there flows a current i proportional to the "weight" of the given bit, is connected to the positive terminal of the 150-v voltage supply. With diode D_1 in the conducting state, current i flows through resistor R_0 and is summed with the currents from the other bit positions at resistor R_0.

Diode D_1 serves as a switch which conducts or is cut off by the voltage from winding a of the dynamic flipflop of the given bit position of the counter. If dynamic flipflop FF is conducting (code 1), then there are negative pulses in winding a of the transformer to cut off diode D_1. With code 0, the dynamic flipflop does not deliver voltage at the output and diode D_1 is in the conducting state. Due to the relatively low reverse resistance of diode D_1 when it is in the cutoff state, a leakage current flows through its reverse resistance, producing a voltage error at resistor R_0. This error is compensated by connecting another winding b in the dynamic flipflop circuit, which provides a voltage of positive polarity. This voltage gives a compensating current which has a polarity opposite that of the leakage current of the diode. The compensating current flows through diodes D_4, D_5, resistor R_2 and resistor R_0. Capacitors

*These values are voltages for a vacuum-tube circuit.

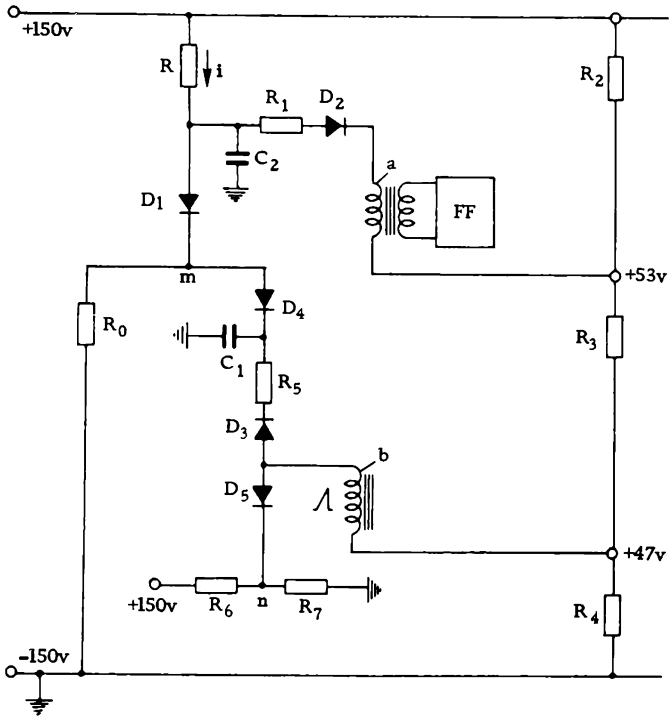


Fig. 221. One bit-position circuit of the converter in Fig. 220.

C_1-C_2 serve to smooth the ripple of the current rectified by the diodes.

As was discussed above, the voltage drop across resistor R_0 (the potential at point m) is automatically held constant to a value of +50 v. With diode D_1 conducting, diode D_2 is cut off by a voltage of +53 v applied to its cathode from the voltage divider consisting of resistors R_2-R_4. When a negative pulse in winding a of the dynamic flipflop cuts off diode D_1, the current flows through diode D_3 and resistors R_3 and R_4 of the voltage divider. A positive voltage pulse in winding b of the dynamic flipflop (through diode D_3), resistor R_5 and the reverse resistance of diode D_4 produces in resistor R_0 a current which compensates the leakage current. In the absence of a pulse in winding b, diode D_3 is cut off by a voltage of +47 v from the divider consisting of resistors R_2-R_4. Diode D_5 diverts the current flowing through R_2-R_4 from winding a. By means of divider R_6-R_7 the potential at point n is held to approximately 49 v. The entire circuit is supplied by a single voltage source of +150 v.

Figure 222 shows the diagram of a 10-bit voltage converter which is composed of the above single-bit circuits. The currents from all bit positions of the converter are summed in resistor

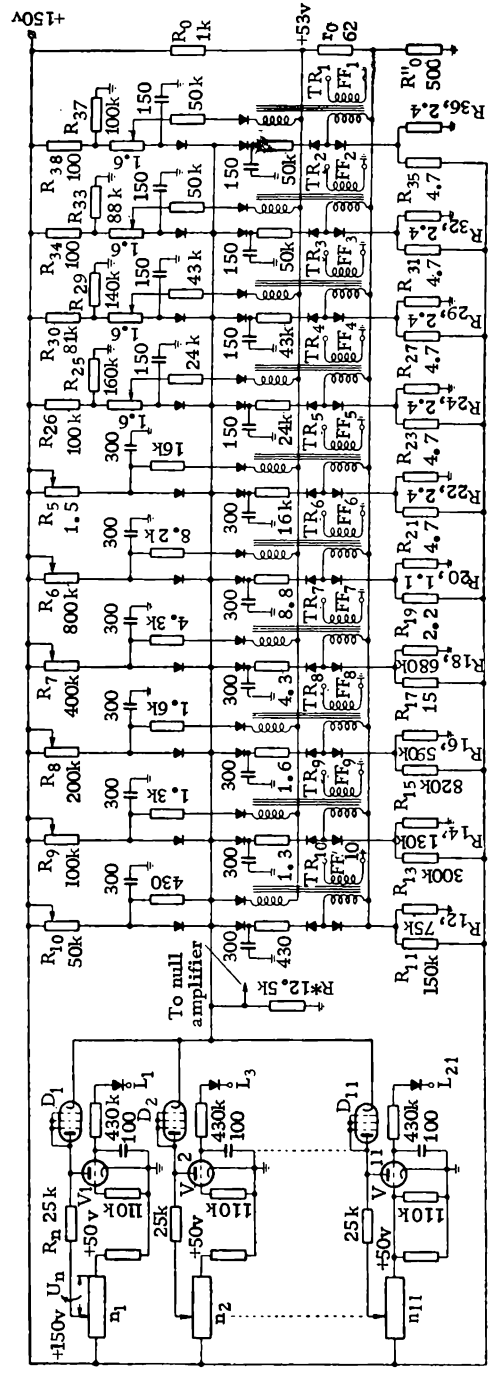


Fig. 222. Diagram of a 10-bit voltage converter.

R_0 = 12.5 kilohms. Resistors R in the circuits for the six most significant bits have values of 50, 100, 200, 400, 800 and 1600 kilohms. For the remaining (least significant) four bits a voltage divider is used in order to decrease the value of resistor R to 1.6 megohms and to obtain therein a current proportional to the weight of the given bit. Control of the diodes passing current from each bit position to resistor R_0 is achieved by voltages from the windings of dynamic flipflops FF_1-FF_{10} of the number code register in the converter. The reference voltages of +53 v and +47 v (applied to the windings of the dynamic flipflops), which are common to all the bit positions of the converter, are provided by a common voltage divider consisting of resistors of 1 kilohm, 62 and 500 ohms. Switching of the voltages from the input potentiometers n_1 to n_{11} is achieved by circuits using triodes V_1-V_{11}. These triodes are normally conducting and thus cut off vacuum-tube diodes D_1-D_{11}. With the application of control voltage to the grid of one of triodes V_1-V_{11}, the triode is cut off, thereby connecting one of the converter channels. The 430-kilohm resistor in the grid circuit of the triode and the 100-$\mu\mu$fd capacitor serve as a filter for the pulse train at one of lines L_1-L_{11} from the control unit of the computer.

2. A CONVERTER USING TRANSISTOR SWITCHES

Figure 223 shows a converter in which the voltage across summing resistor R_0 depends on the sum of branch currents. In distinction from the previous circuit, the current from the potentiometer transducer in this converter does not pass through the common resistor R_0 and the voltage drop across resistor R_0 depends only on the code in the register. Thus, we have a device for conversion of a digital code to a voltage. Transistors T_1-T_n are used for switching of currents in the individual branches in accordance with the code in the register.

The sum of the currents flowing through the common resistor is

$$I_\Sigma = \sum_{h=2^0}^{h=2^n} h \cdot i \cdot K, \qquad (3)$$

where K is 0 or 1 respectively for code 0 or 1 in the flipflop stage of the register. It is obvious that the total current I_Σ in this circuit and the output voltage U_{out} is not exactly proportional to the binary number in the register. With disturbance of this proportionality an error occurs, with the result that the potential at point m does not remain constant with branching of the currents according to Kirchoff's law. Since for practical purposes the value of summing resistor R_0 cannot be extremely small, a conversion error occurs when more than one transistor conducts simultaneously; this error reaches a maximum when all transistors are in the conducting state. However, with appropriate selection of currents and resistances,

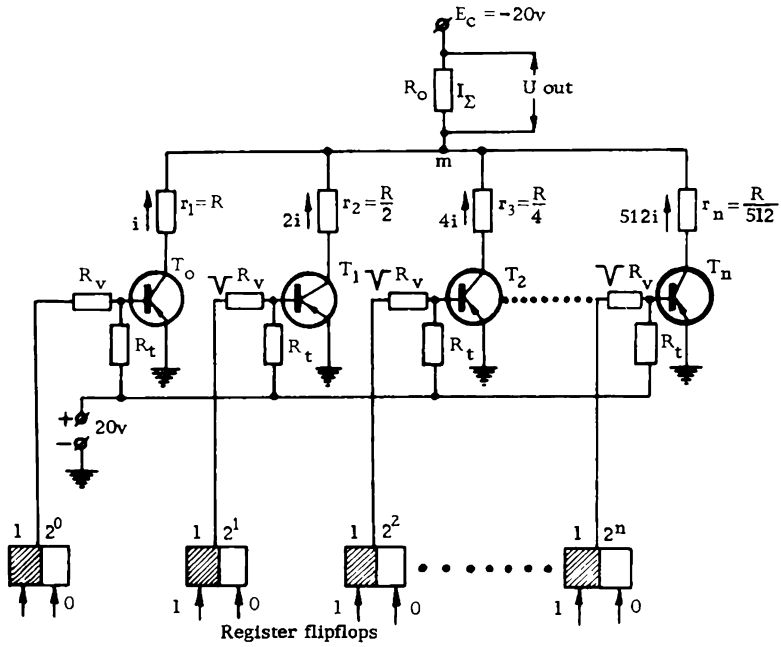


Fig. 223. Transistor converter with current summing.

this error may be held to the order of 0.1% for a limited number of stages in the register (10 bit positions). The currents and resistances may be calculated from the following formulas:

Total current

$$I_\Sigma = \sum_{n=1}^{n} 2^{(n-1)} \cdot i, \tag{4}$$

where i is the minimum current in the first branch; n is the number of current branches (number of bits).

The resistance of each branch is

$$r_n + R_0 = \frac{E_S}{2^{n-1} \cdot i} \tag{5}$$

or

$$r_n = \frac{E_S - 2^{n-1} \cdot i \cdot R_0}{2^{n-1} \cdot i}, \tag{6}$$

where E_S is the supply voltage.

The total parallel resistance R_n of all branches

$$R_n = \frac{1}{\sum_{n=1}^{n} \left[\frac{2^{n-1} \cdot i}{E_S - 2^{n-1} \cdot i \cdot R_0} \right]}. \tag{7}$$

The maximum error occurs when all transistors are in the conducting state and is

$$\delta_{max} = \frac{R_0}{R_n + R_0} \times 100\%. \quad (8)$$

An example of the maximum error δ_{max} calculated for different values of summing resistor R_0 for 10 bits is given in Table 13.

TABLE 13

Converter Errors for Different Values of Current in Summing Resistor.

Supply voltage, E_s (volts)	Minimum current i (amperes)	Error δ_{max} in %		
		R_0 = 1 ohm	R_0 = 10 ohms	R_0 = 100 ohms
20	$1 \cdot 10^{-6}$	0.00511	0.0511	0.510
20	$10 \cdot 10^{-6}$	0.0511	0.510	4.95
20	$25 \cdot 10^{-6}$	0.128	1.27	11.8
20	$50 \cdot 10^{-6}$	0.255	2.51	21.9
20	$100 \cdot 10^{-6}$	0.510	4.95	38.4
25	$1 \cdot 10^{-6}$	0.0409	0.0409	0.408
25	$10 \cdot 10^{-6}$	0.0409	0.408	2.98
25	$25 \cdot 10^{-6}$	0.102	1.016	9.58
25	$50 \cdot 10^{-6}$	0.204	2.019	18.03
25	$100 \cdot 10^{-6}$	0.408	3.984	32.30

It is seen from Table 13 that the error for the case in which the minimum current is 10 μa and the summing resistor is 1 ohm and the case in which the current is 1 μa and the summing resistor is 10 ohms is less than 0.1%. An increase in supply voltage permits an increase in the summing resistance, while an increase in the ratio $\frac{r_n}{R_0}$ decreases the error. The minimum output voltage U_{out} for these cases is only 10 μv and an extremely sensitive null amplifier is required. This is an important disadvantage. An increase in output voltage at the summing resistor may be achieved only by the use of a higher supply voltage or larger currents.

The transistor is not an ideal current switch, because there is a leakage current which increases rapidly with temperature. In order to compensate the leakage current, a positive voltage source (+20 v) is connected to the base-emitter circuit, cutting off the transistor and causing a reverse base current. This current, which

compensates the effect of the leakage current, is chosen approximately equal to the leakage current of the collector junction at the upper limit of the temperature range. With a positive bias at the base, a considerably higher amplitude of control signal is required to cut off the transistor. Resistances R_v and R_t in each branch are so chosen as to ensure compensation of the leakage current. A small part of each of resistors $r_1, r_2, \ldots, r_n$ is made adjustable so as to achieve precise matching. These resistors must be made with high precision, particularly those for the more significant bits, and must be stable for temperature changes. The error of the above described inverter circuit using silicon transistors may not exceed 0.1% for 10 bits with small changes in temperature. In comparing this circuit (Fig. 223) with the previously described converter (Fig. 222), this circuit delivers an effective signal 1000 times smaller than that of the previous circuit.

3. A DIGITAL POTENTIOMETER

The digital potentiometer, known as a scale-factor potentiometer, gives an output voltage proportional to a given digital binary code. It is essentially a device for conversion of a digital code to a voltage. In the collector circuits of the register flipflops the windings of electromagnetic relays $P_1, P_2, \ldots, P_n$ (not shown in the diagram) are connected so that presence and absence of current in the relay windings establishes code 1 and code 0, respectively. Each relay, corresponding to a given bit, has a pair of normally closed and normally opened contacts. Upon closure of the relay, the normally closed contacts connect precision resistors $R_0, R_1, R_2, \ldots, R_n$ in the circuit. The values of these resistors correspond to the weights of the bit positions which the resistors represent (i.e., the weights of $2^0, 2^1, 2^2, \ldots, 2^{n-1}$, where n is the bit position). In order to maintain constant current in the potentiometer circuit (which is fed from voltage source E), the normally opened contacts of these relays short out resistors $R'_0, R'_1, R'_2, \ldots, R'_{n-1}$ upon closing. In this way, the output voltage U_{out} of the potentiometer is proportional to the binary number stored in the register. The accuracy of this type of digital potentiometer, depending on the number of bit positions of the register, can be quite high. If potentiometer supply voltage E is varied in proportion to a quantity x, the potentiometer may be used for multiplication of two quantities $x \cdot y$, where x is a continuous voltage and y a digital code.

A disadvantage of the digital potentiometer in which relay contacts are used is low conversion speed, as the relay can operate at a time interval no smaller than a few milliseconds. If the relay contacts are replaced by pulse-operated transistor switches (Fig. 225), the digital potentiometer can achieve a high conversion speed. Transistors $T_0, T_1, \ldots, T_{n-1}$ in the presence of input pulses are either conducting or cut off, depending on whether the corresponding bit positions of register P are of code 0 or 1, respectively. Conversely, in the presence of pulses transistors T'_0, T'_1, T'_{n-1} are

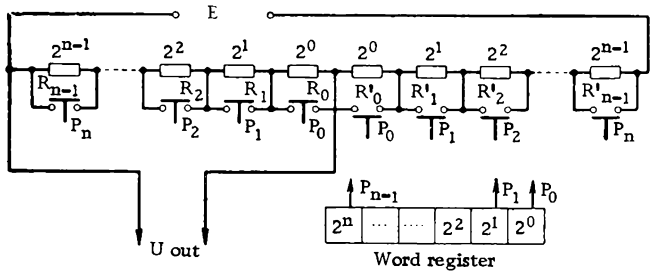


Fig. 224. A digital potentiometer using relay switches.

cut off for code 0 or conduct for code 1 in the corresponding register stages. For this purpose, each stage of register P, consisting of dynamic flipflops, has two outputs: one corresponding to code 0 and the other to code 1. When a flipflop is in the zero state, pulses occur at the code 0 output; conversely, when the flipflop is in the 1 state, pulses appear at the code 1 output. If all bit positions of register P are in the 0 state, all transistors $T_0, T_1, \ldots, T_{n-1}$ conduct, while all transistors $T'_0, T'_1, \ldots, T'_{n-1}$ are cut off when the CP pulses are applied to the dynamic flipflops. Thus, at the moment of application of a CP pulse the voltage U_{out} is zero. For code 1 in all register positions, conversely, all transistors $T_0, T_1, \ldots, T_{n-1}$ are cut off, while transistors $T'_0, T'_1, \ldots, T'_{n-1}$ conduct; the voltage U_{out} at the moment of application of a CP pulse is equal

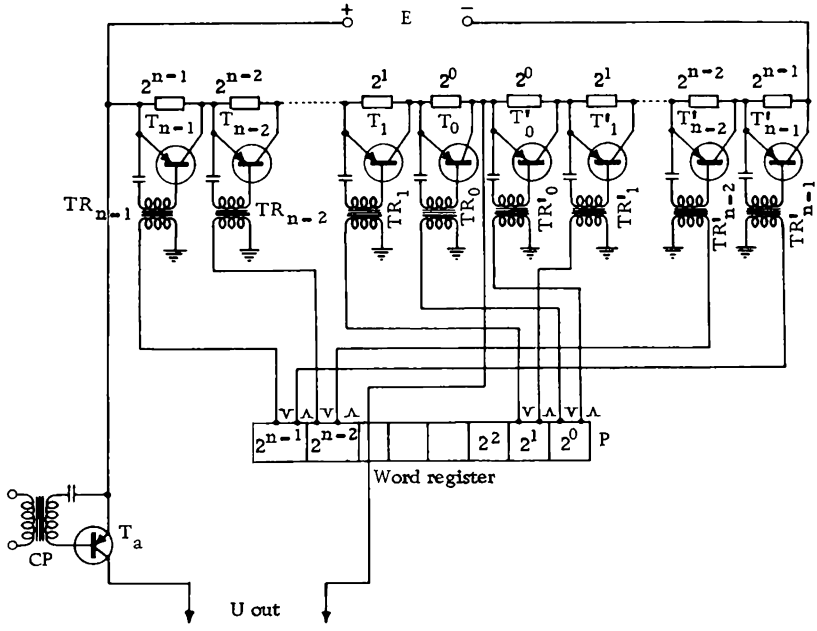


Fig. 225. A digital potentiometer using transistor switches.

to the supply voltage E. In order to control the voltage U_{out}, transistor switch T_a is used; this switch is driven to conduct by CP pulses. The accuracy of the potentiometer with transistor switches depends on the emitter-collector resistance of the cut-off transistor, which shunts a corresponding potentiometer resistor, and on the forward resistance of the conducting transistor (which is not equal to zero and consequently does not completely short out the corresponding potentiometer resistor). By selecting sufficiently large potentiometer resistances and by error compensation methods, the potentiometer error can be decreased. A more accurate digital potentiometer is one having circuits for coarse and fine digitization. Digital potentiometers may be used in computers of hybrid type containing both analog and digital computer elements. Hybrid elements may also be used with success in control systems requiring data interchange between an analog computer and various digital transducers. The use of the latter is dictated by the necessity of decreasing the error in transducer readings as well as by the convenience of digital telemetering. The computation may be achieved by analog techniques when high accuracy is not required and when there is no need of a memory. In this case, the analog computer may receive information in digital form and, by means of a digital potentiometer, may use the information in the computing element (e.g., multiplication, squaring, extracting roots, connection of the digital potentiometer in a Wheatstone bridge, etc.). Such methods of inserting digital information into an analog computer may prove to be more effective than direct conversion of digital data to analog data.

4. A MULTICHANNEL CONVERTER USING A DIGITAL POTENTIOMETER

The converter described below for conversion of a voltage to a digital code and vice versa is based on the use of a digital potentiometer which is switched in accordance with the digital code stored in the register.

The block diagram of the converter is given in Fig. 226. The converter is constructed in the form of a bridge. One pair of the bridge arms consists of an electronic voltage divider. The "slider" of the electronic divider is set according to the number code stored in a register. The other pair of arms contains transducer potentiometers which are alternatively connected to the bridge. These potentiometers are constantly connected to the supply source and, during conversion, the voltages at the sliders are alternatively applied to the input of null amplifier NA by means of switches K_1, $K_2, \ldots, K_n$ such as in the circuit described in Section 5. The voltage is converted to a code by successive approximations. First, the most significant bit position in the register is set to the 1 state. The converter and transducer are then interrogated. Thereupon the digital potentiometer delivers a voltage corresponding to the code

in the register. The null amplifier determines the sign of the inequality between the voltages from the transducer and the digital potentiometer. If the latter is equal to or larger than the former, code 1 in the most significant bit position of the register remains. If the latter is less than the former, code 1 in the most significant bit position of the register is changed to code 0. Code 1 is then set in the next most significant bit position and the interrogation cycle (or comparison cycle) is repeated. After m such cycles (m being the number of bits), the code in the register gives the digital value of the converted voltage.

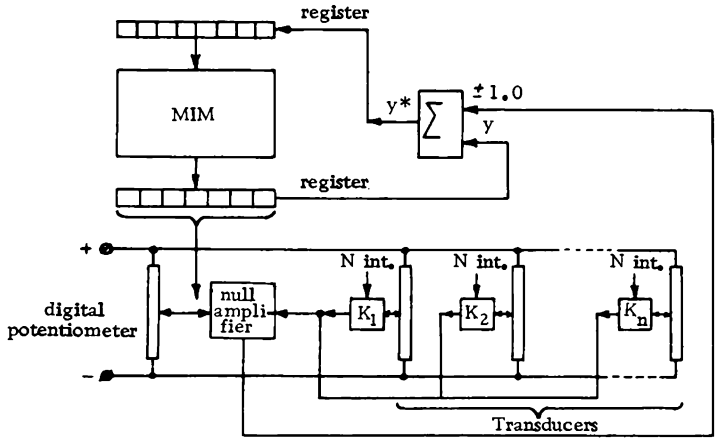


Fig. 226. Block diagram of a converter using a digital potentiometer.

The above circuit is convenient for data entry into a digital integrating computer. The null amplifier generates the increment of the input variable in an iteration, when the value in the current iteration is compared with that in the preceding iteration. The code corresponding to the input variable in the previous iteration is stored in the memory and is transferred to the converter register at the time of interrogation or comparison of the given input channel.

The bridge circuit permits the use of an unregulated power supply, which is particularly important in those cases where the transducers are located some distance apart. Another advantage of the bridge circuit is the currentless condition of the switches in the balance state. In the presence of a large unbalance, a large current flows along the diagonal of the bridge and conversion accuracy may become low. Maximum accuracy of the circuit is required when the bridge is balanced (i.e., no current in the diagonal). This permits use of a null amplifier with a low input resistance.

Accuracy of the entire conversion depends on the accuracy of the digital potentiometer. The "quantum" of the conversion is determined by the required conversion accuracy. For a 10-bit converter, the quantum is approximately 0.001. In order to obtain

such a small quantum, the digital potentiometer consists of two stages: coarse and fine. The coarse stage is controlled by a few of the most significant bits and the fine stage by the remaining least significant bits. The coarse stage serves to connect the fine stage to a voltage interval, depending on the values of the most significant bits.

Figure 227 shows the diagram of the coarse stage controlled by the three most significant bits of the code. The voltage range is divided into eight equal intervals by eight equal resistors $R_0 = R_1 = R_2 = R_3 = \ldots R_7 = R$ connected in series. The switches (K's) are operated in such manner that, depending on the code of the most significant bits, they connect the fine stage in parallel with the chosen resistor R. For example, if the three most significant bits contain the code 000, switches K_0 and K_0' are closed and the fine stage is connected in parallel with resistor R_0. With code 011, switches K_3 and K_3' are closed and the fine stage is connected in parallel with resistor R_3. With the maximum code 111, switches K_7 and K_7' are closed and the fine stage is connected in parallel with resistor R_7.

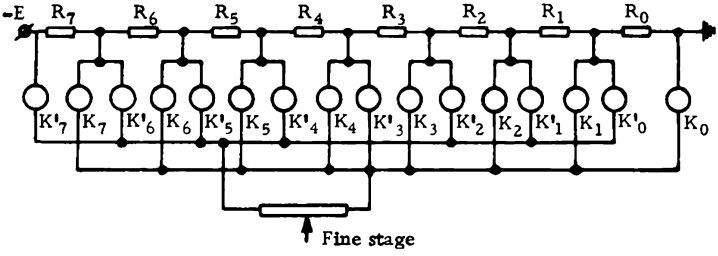


Fig. 227. Coarse stage of a digital potentiometer.

The ratio of the resistance R of the coarse stage and the resistance of the fine stage must be such that the current of the fine stage does not distort the potential in the coarse stage. Fulfillment of this requirement is not obligatory in using the compensation method discussed below.

Let us discuss the operating conditions of switches at the extreme positions of the "slider" of the coarse stage. In the presence of code 000 switches K_0 and K_0' are closed. Switches K_1 to K_7 and K_1' to K_7' are then acted upon by a positive reverse voltage. In the presence of code 111 switches K_0 to K_6 and K_0' to K_6' are acted upon by a negative reverse voltage. Thus, depending on the code, reverse voltage of different polarities is applied to switches K_1 to K_6 and K_1' to K_6'. Hence the switches must be bipolar.

Figure 228a shows the diagram of a coarse stage which permits the use of unipolar switches. The circuit is based on the fact that the coarse stage is actually not a voltage divider but a special switch which delivers to the fine stage the required voltages according to the code in the register. These voltages are created by individual

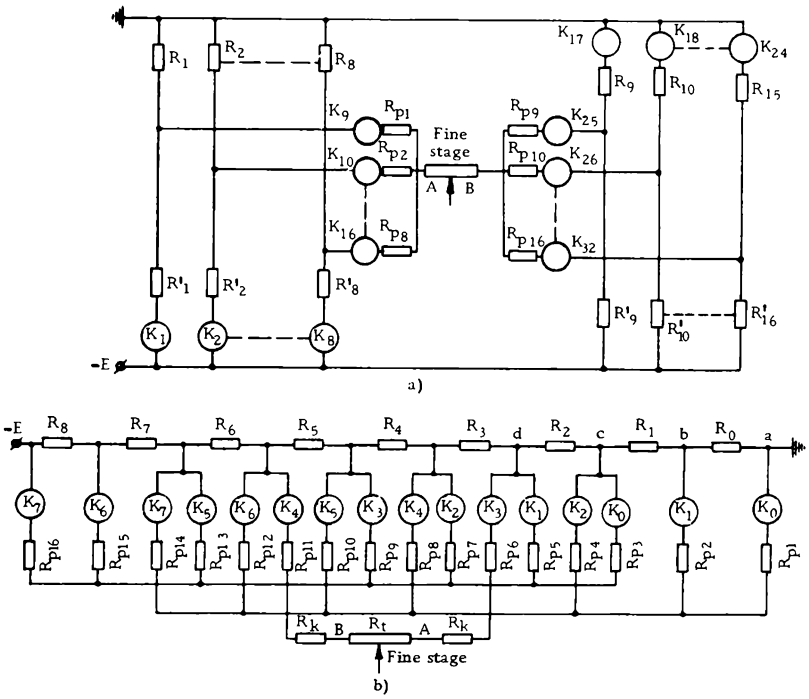


Fig. 228. Diagram of coarse stage:
a—with unipolar switches; b—with bipolar switches.

dividers, each voltage interval of the coarse stage requiring two dividers: one for application of voltage at point A of the fine stage by the dividers consisting of resistors R_1 to R_8 and R'_1 to R'_8; and the other for application of voltage at point B by the dividers consisting of resistors R_9 to R_{16} and R'_9 to R'_{16}. Connection of the fine stage to the lefthand group of dividers is achieved by switches K_9 to K_{16}, and to the righthand group by K_{25} to K_{32}. In order to use unipolar divider switches, the voltages are also connected to switches K_1 to K_8 and K_{17} to K_{24}. It is apparent that, regardless of the code, the open switches are always acted upon by a reverse voltage of the same polarity

The fine stage of the digital potentiometer may make use of one of the previously discussed circuits. Actual electronic switches in the "ON" state have a forward resistance $R_f > 0$ and in the "OFF" state a reverse resistance $R_r < \infty$. These two resistances account for the errors which the switches introduce into the converter. A method for compensating these errors assumes that the function of the coarse stage is to deliver a selected potential to the input of the fine stage.

Figure 228b shows the compensation method for the coarse stage with bipolar switches. The fine stage is connected not to

one resistor, but in parallel with two adjacent resistors. For example, in the presence of code 000 switches K_0 and K_0' are closed and the fine stage is connected to points a and c. For code 001, switches K_1 and K_1' are closed and the fine stage is connected to points b and d. Resistors R_* and R_p in each branch are so chosen that the required potentials are established at points A and B. The potential at point A is

$$U_A = -\left(\frac{E \cdot K}{n_c + 1} + \frac{1}{2}\frac{E}{n_c + 1}\right). \tag{9}$$

and the potential at point B is

$$U_B = -\left(\frac{E \cdot K}{n_c + 1} + \frac{3}{2}\frac{E}{n_c + 1}\right), \tag{10}$$

where K is the decimal value of the code (for code 011, $K = 3$), n_c is the number of discrete steps in the coarse stage of the converter, and E is the supply voltage.

It is not difficult to see that, depending on the code, the potentials across points A and B lie midway between the corresponding resistors and that the voltage applied to the fine stage is always equal to the voltage drop across R_t. In order to avoid loss of voltage range of the transducers, two resistors R_k are connected in series with the latter being potentiometers.

Compensation in the circuit with unipolar switches makes use of the same technique. As shown in Fig. 228a, compensating resistors R_{p1}, R_{p2}, ..., R_{p16} establish the appropriate potentials across points A and B for the fine stage in accordance with the values of E and n_c.

The above compensation method permits the finite forward and reverse resistances of the switches as well as tolerances in manufacture of the divider resistances. Any change in circuit parameters after adjustment will lead to errors in converter operation.

Transistors operating at deep saturation condition may be used as unipolar switches in voltage dividers. Figure 229 shows the characteristics of P11 and P16A transistors $U_{ce} = f(i_b)$ for different collector currents. The voltage between the collector and emitter U_{ce}, beginning with a sufficiently large base current i_b, is independent of i_b for a given collector current. Selection of the base current i_b is made from the characteristics $U_{be} = f(i_b)$.

The null amplifier (NA) determines the sign of the voltage difference between the digital potentiometer and the transducer. One of the NA inputs is connected to the digital potentiometer, the second input to the transducer, and the third input to the source of the "interrogate" pulse.

The block diagram of a null amplifier is shown in Fig. 230a and the schematic in Fig. 230b. There are two identical channels in the null amplifier. Each channel contains a two-stage amplifier, a coincidence circuit and a blocking oscillator. The inputs to the null

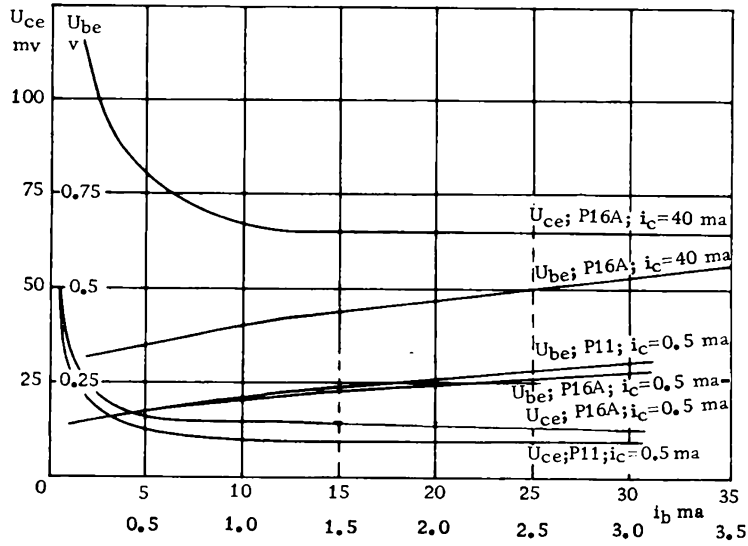


Fig. 229. Characteristics of transistors P11 and 16A.

amplifier are connected to the primary winding of a transformer. The two ends of the secondary winding are connected to the two channel amplifiers, the center of the secondary winding being grounded. In the presence of an unbalance, signals of opposite phases are applied to the two amplifiers. In the presence of a negative signal at one of the channel inputs in excess of the threshold value, a negative pulse is applied from the amplifier to the coincidence circuit of this channel. The coincidence circuit is strobed by a pulse with a duration of 0.5 μsec and with a 4-5-μsec shift from the start of interrogation. This shift is sufficiently long to damp the transients caused by the "interrogate" pulse and then to receive the unbalance voltage of the bridge circuit. A pulse at the coincidence circuit output triggers the blocking oscillator. Operation of the blocking oscillator indicates the presence of an unbalance.

The transient settling time of the input transformer of the null amplifier is important to the operating speed, particularly when the interval between the "interrogate" pulses is short. If a large unbalance is applied to the null amplifier during the previous interrogation, the response during the present interrogation is distorted due to the unsettled transient. In order to damp the transients, two transistors are shunted across the primary winding of the input transformer. During the interval between interrogations these transistors are in the conducting state. During interrogation a cut-off pulse is applied to the transistor bases. In the presence of a large unbalance one of the transistor collectors becomes forward biased and limits the unbalance amplitude. This limited signal is sufficient to drive the null amplifier without causing extreme saturation.

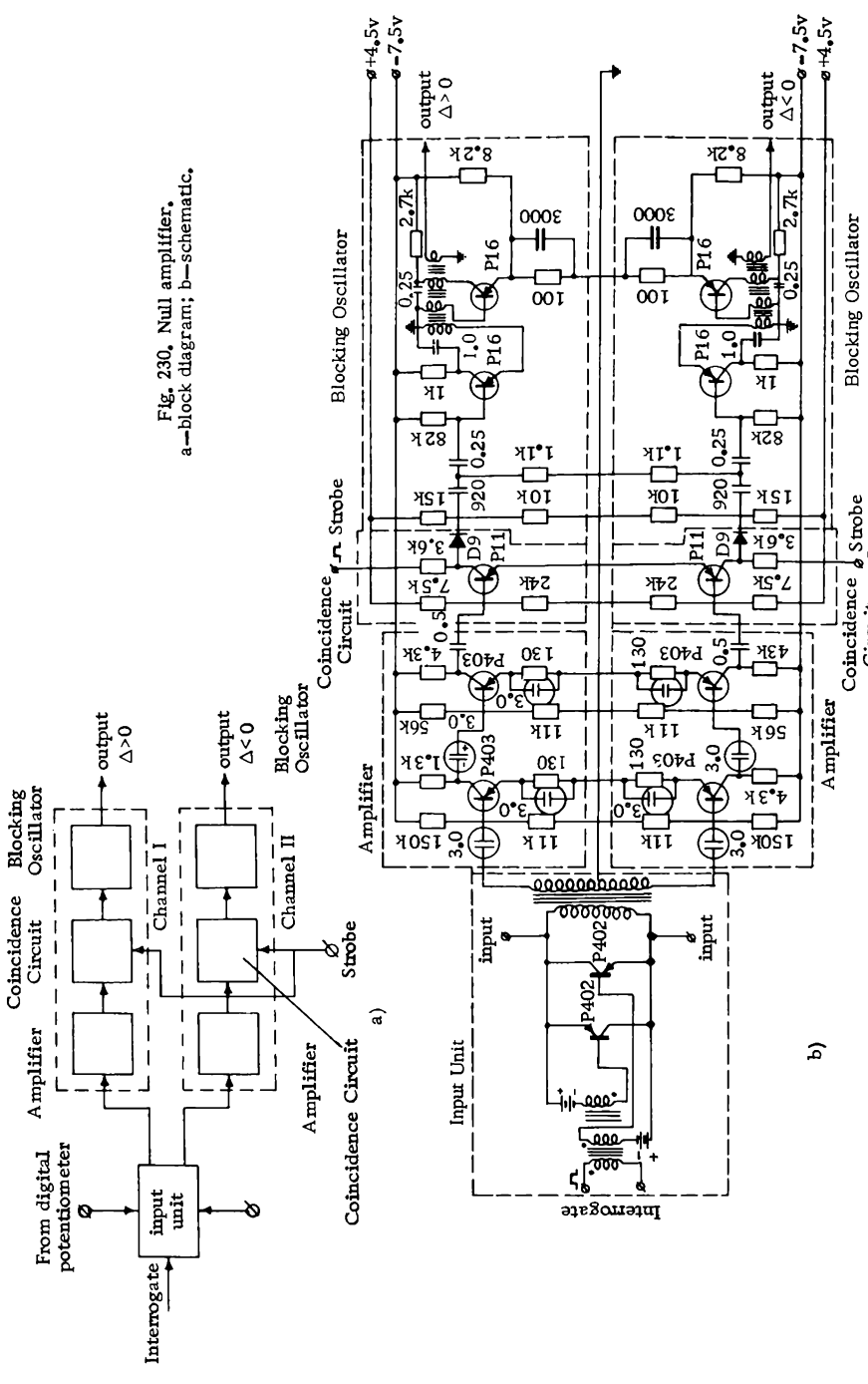


Fig. 230. Null amplifier.
a—block diagram; b—schematic.

The amplifier must cause little stretching of the input pulses since, due to the differences in the edges of the two input pulses, in the presence of an unbalance close to zero fluctuations occur during the damping time of 1-1.5 μsec. In order to decrease turn-off time of the transistors of the amplifiers, compensating RC networks are connected in the emitter circuits and proper collector voltages are chosen.

The coincidence circuit uses a P11* transistor (n-p-n) which is normally operated in the saturation region. Upon application of a negative pulse at the base the transistor is cut off, the collector-emitter resistance rises sharply, and the strobe pulse triggers the blocking oscillator. Sensitivity of the null amplifier is 7-10 mv.

The converter control unit shown in Fig. 231 consists of the following circuits:

a) fine stage control;
b) coarse stage control;
c) commutator control;
d) input variable commutator;
e) converter register.

The fine stage control circuit (without error compensation) includes eight inhibit circuits and as many coincidence circuits. Through one of the inputs of each circuit, these circuits are set up by the converter register. Depending on the state of the flip-flop of the given bit position, upon the application of an "interrogate" pulse a coincidence circuit or an inhibit circuit is triggered If a coincidence circuit is triggered, the resistor in one arm of the digital potentiometer is shorted out, while upon triggering of an inhibit circuit the corresponding resistor in the other arm is shorted out. The coincidence circuit is based on the usual diode-transformer principle without amplification and the secondary winding of the transformer directly controls the shunt transistor. The inhibit operation is achieved by means of a diode circuit with subsequent amplification and shaping of the signal.

The coarse stage control circuit decodes the state of the three most significant bits of the converter register. The code of flip-flops 9 and 10 is decoded by a diode decoder. Since the flipflops of the converter register do not have paraphase outputs, there are two inverters at the input of the diode decoder. The four outputs of the diode decoder are applied to the horizontal lines of the coarse-stage diode-transformer matrix through emitter followers.

Depending on the state of flipflop 11 of the converter register, the "interrogate" pulse passes through a coincidence or an inhibit circuit to one of the vertical lines of the coarse-stage matrix. Each of the matrix transformers has four secondary windings. The output line of each matrix transformer is connected to two corresponding coarse-stage dividers (one on the right and one on the left) which are connected to the fine stage.

---

*A better choice would be the P406 transistor, having an alloyed junction and a lower voltage drop at the collector-emitter junction.

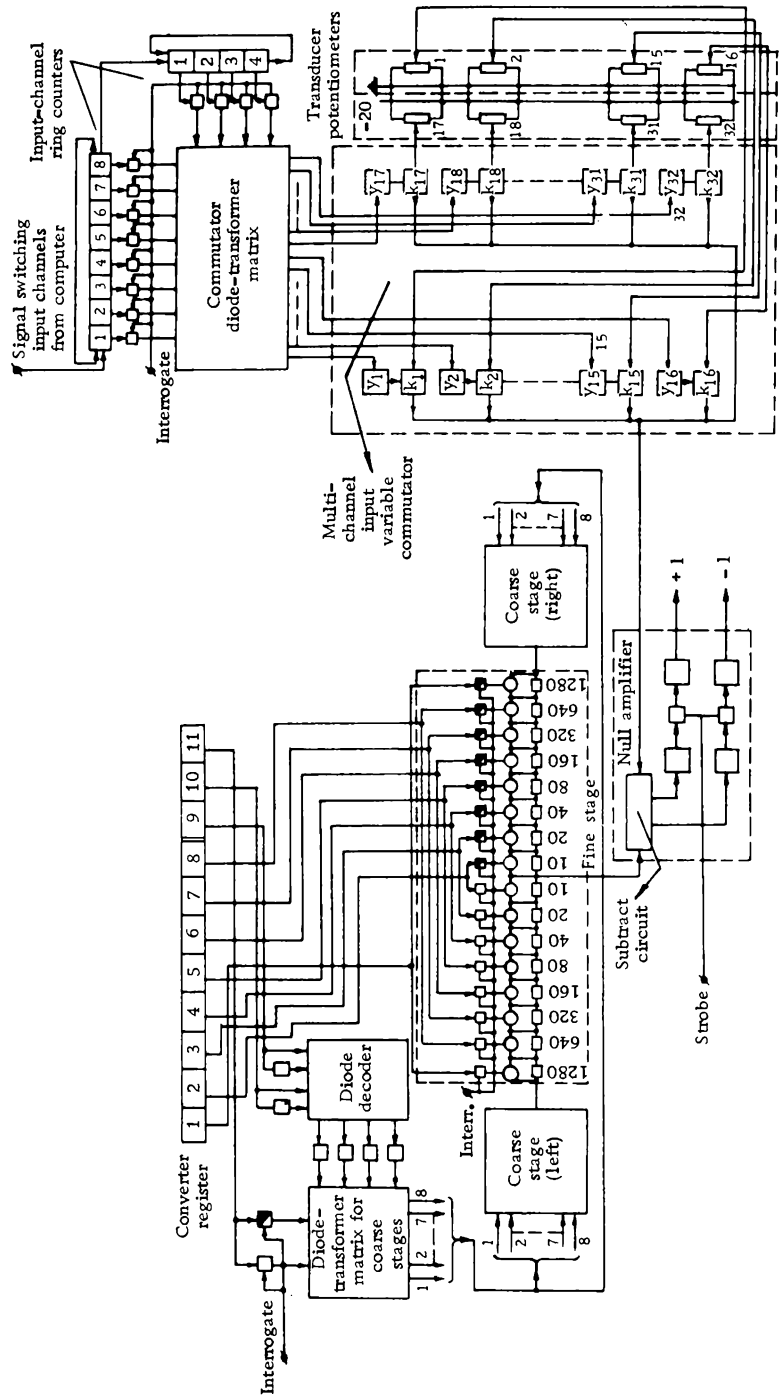


Fig. 231. Block diagram of converter.

The commutator control circuit is a diode-transformer matrix with $4 \times 8 = 32$ outputs. The vertical and horizontal lines of the matrix receive signals from coincidence circuits controlled by two ring counters which are switched by a signal from the computer. At a given moment, only one flipflop of the counter is in the 1 state, thereby selecting one output line of the commutator diode-transformer matrix. The converter register consists of dynamic flip-flops with voltage outputs.

5. A MULTICHANNEL TRANSISTOR SWITCH

In linking a control computer with a controlled process the input variables are often represented in the form of voltages; thereby a multichannel voltage-to-code converter is required. Since electronic converters are rather expensive and complex, a single converter is time-shared so that the voltages of many input data transducers are alternatively applied to the converter. These voltages are switched by a precision multichannel switch.

Figures 232a and b, respectively, show the block diagram of a multichannel switch and the schematic of a single switch. In Fig.

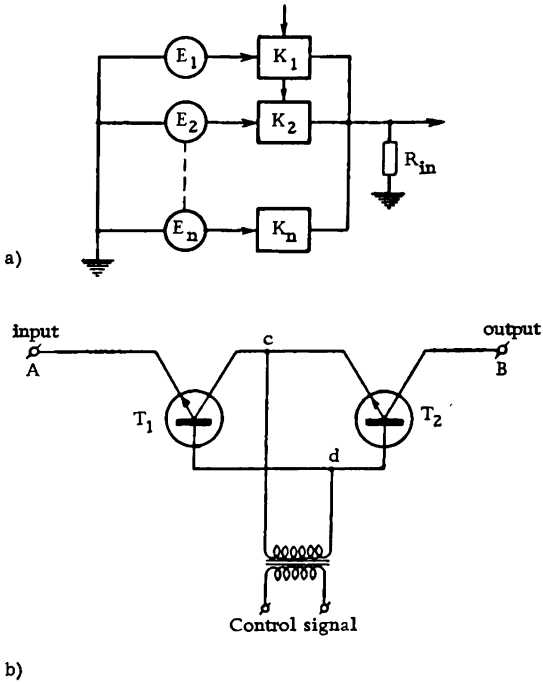


Fig. 232. A multichannel switch:
a—block diagram; b—schematic of a single switch.

232a, E_1, E_2, ..., E_n are the voltage sources, which are connected in a certain time sequence by switches K_1, K_2, ..., K_n to a common input; R_{in} is the input resistance of the succeeding circuit. In Fig. 232b, in the presence of a large number of voltage sources, silicon transistors are used for maximum isolation of the source voltages. The switch transistors are operated with reverse connection (i.e., the emitters serve as collectors and vice versa), which decreases the influence of the control signal on the operating current of the switch. In switching operation, the switch has a high resistance when it is cut off and a low resistance when in saturation. When the input signal E is of a positive or a negative polarity, the collector of one transistor is connected in the reverse direction and that of the other in the forward direction; this arrangement permits transmission of a voltage of either polarity.

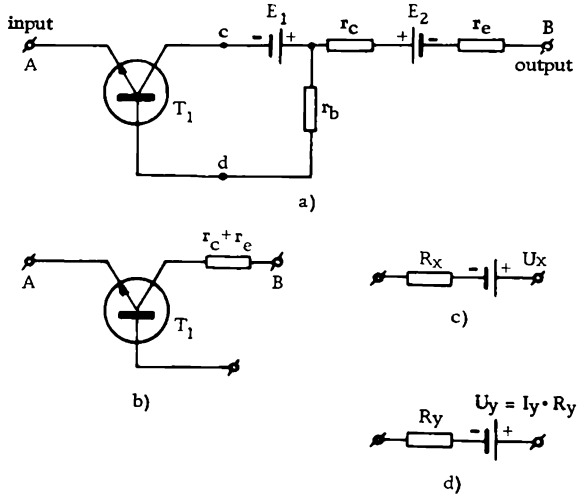


Fig. 233. Equivalent circuits of the switch:
a—equivalent circuit of transistor T; b—simplification of a; c, d—equivalent circuits for ON and OFF conditions, respectively.

For analysis of the transistor operation in the switch, assume that the input terminal of the switch is at a higher potential than the output terminal and that a control pulse with amplitude sufficient to drive the transistors to saturation is applied to points c and d. The n-p-n transistor T_2, whose collector is forward biased, may be represented by the equivalent circuit in Fig. 233a. In a silicon transistor, $r_b \gg r_e \approx r_c$; the equivalent circuit in Fig. 233a is reduced to that in Fig. 233b. The switch may be investigated on the basis of large-signal operation of transistor T_1 in the common-emitter configuration (with consideration of the reverse connection). In this case, the transistor may be replaced by two equivalent circuits corresponding to the ON and OFF conditions as shown in Fig. 233c and Fig. 233d, respectively.

In these circuits

$$R_x = R_{cx} + R_{ex} + \frac{\delta}{I_1}\left(\frac{\alpha_i}{\alpha_n}\right)(1 - \alpha_i \cdot \alpha_n), \tag{1}$$

$$U_x = I_1 \cdot R_{ex}(1 - \alpha_i) \cdot \delta, \tag{2}$$

$$\frac{1}{R_y} = \frac{1}{R_{c\,in}} + \frac{I_{cS}}{\delta} \cdot e^{-\left(\frac{-E_1}{\delta}\right)}, \tag{3}$$

$$I_y = -\frac{E_1}{R_{c\,in}} + (1 - \alpha_i) I_{cS}\ , \tag{4}$$

where $R_{c\,in}$ is the leakage resistance;
R_{cx} and R_{ex} are the ohmic resistances of collector and emitter, respectively;
α_r is the current gain in a reverse connection;
α_f is the current gain in a forward connection;

$$\delta = \frac{KT}{q},$$

where K is the Boltzmann constant;
T is absolute temperature in degrees Kelvin;
q is the electron charge;
$\frac{KT}{q} \approx 26$ mv at 25° C;
I_1 is the operating current of the switch, determined by the transducer voltage;
I_{cS} is the emitter current with emitter as collector.

It is seen from the above formulas that the accuracy of voltage transfer by the switch is determined by many transistor parameters. In view of the wide spread of these parameters in production, it is difficult to select transistors on the basis of any one parameter. It is recommended that the selection be made experimentally for the specific switching circuit.

The precision with which the voltage is transmitted by the switch depends substantially on the input resistance R_{in} of the succeeding circuit (Fig. 232a). The value of R_{in} must be considerably larger than the forward resistance R_f of the switch. For silicon transistors, resistance R_f is of the order of several hundred ohms. In order to ensure sufficient accuracy in voltage transfer regardless of the individual parameters of the switch, R_{in} must have a value of hundreds of kilohms. Hence it is preferred that the input circuit employ vacuum tubes rather than transistors. Since the transistor is a substantially nonlinear element, there is a limit in transfer of the voltages and this limit depends largely on the value of R_{in}.

6. A SHAFT-POSITION CONVERTER

The angular position of a shaft can be converted to a digital code by a coding disk or a cylinder mechanically connected to the shaft.

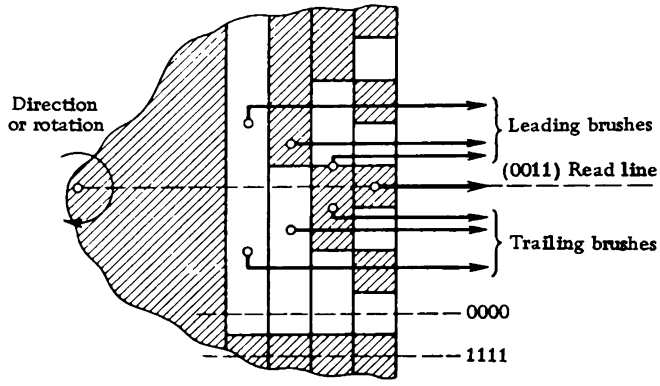


Fig. 234. Part of a coding disk with brushes.

The position of a coding disk can be read out by means of an inductive contact device or a photoelectric device. Of many possible schemes for angular position conversion, let us discuss the scheme using a binary coding disk with contact brushes. The same principle may be applied for readout from contactless converters.

Figure 234 shows part of a code disk with brushes. The disk is divided into as many concentric circular bands as the number of bits of the digital code. Each band is constituted of alternate segments of current-conducting contact and insulation to represent one bit position. The reading of these bit positions along a read radius gives the binary code of the shaft position. The least signigicant bit of the binary code is located on the outermost circular band, while the most significant bit is located on the innermost circular band.

Angular position of the shaft in a digital form is read out by means of brushes positioned along the read radius of the disk. There are one or two contact brushes in each circular band. In readout of the angular position, reading ambiguity may occur on the boundary between two adjacent codes which differ by more than one bit change (change from a contact segment to an insulation segment or vice versa). Such an error may occur due to inaccurate positioning of the contacts and brushes.

In order to eliminate this ambiguity, the reading of all bit positions except the least significant bit is achieved by means of two brushes for each bit position (one brush on each side of the read line). The distance between the two brushes in each bit position is equal to the segment width of the next less significant bit. The elimination of this readout error is based on a property of the numerical order of ordinary binary numbers. When the disk is rotating so that the binary numbers read out from the read line are in an increasing numerical order, a change of the least significant bit from 0 to 1 is not accompanied by such a change in other bit positions. On the other hand, when there is a change

from 1 to 0 in the least significant bit, there must occur a change from 0 to 1 or from 1 to 0 in the adjacent more significant bit position.

Figure 235 shows the schematic of a coding disk converter. For example, consider the case when the digital code 0011 is being read out from the disk. If the least significant bit position of the disk is a code 1 (a conducting sement on the read line), then the next more significant bit position is read out by the trailing brush, which at this time is unambiguously located (code 1 in this example).

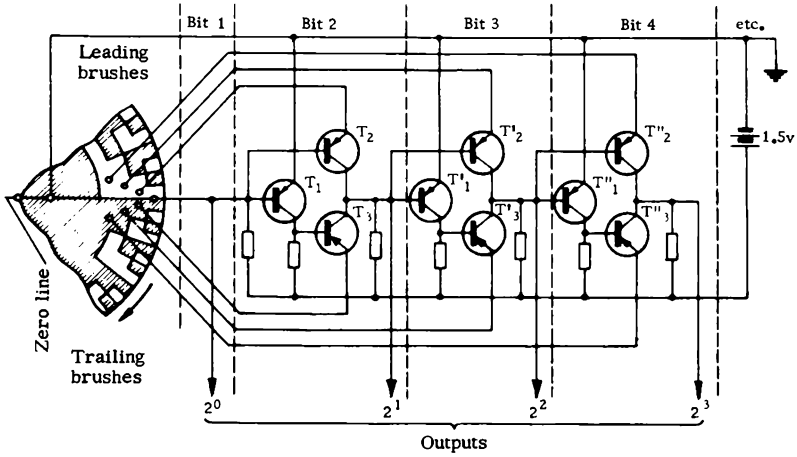


Fig. 235. Diagram of angular displacement converter.

By virtue of the above-mentioned property of binary code, the second bit position does not undergo a change of code when there is a change from 0 to 1 in the least significant bit position.

The third bit position is also read out from the trailing brush, since code 1 is in the second bit position and code 0 in the third bit position. The fourth bit position is read out from the leading brush and we obtain code 0. Thus, the reading of the four bit positions of the disk gives the code 0011. In general, selection of which brush to read out at a given bit position depends upon whether the code in the previous less significant bit is 1 or 0.

Figure 235 shows a transistor switching circuit for brush switching. The first bit is read out directly from the first bit brush. Each of the remaining bits is read out by selecting the proper brush at the given bit position. This circuit consists of three switches. Each switch has three junction transistors, two inputs, and one output. Both inputs of the switch at a bit position are connected to the output of the previous bit position.

For example, the brush switch for the second bit consists of transistors T_1-T_3. Normally, in the absence of external signals, transistors T_1 and T_2 conduct since their bases are negatively biased, while transistor T_3 is cut off since its base is connected

to the collector of conducting transistor T_1. The state of transistor T_1 is determined by code 0 or 1 in the previous bit (the first bit in this example). Transistor T_1, in turn, determines which one of transistors T_2 and T_3 must be conducting (i.e., brush selection). If code 0 is in the least significant bit (or first bit) position, the next bit position must be read out from the leading brush, which is switched in by transistor T_2. Code 0 in the first bit position corresponds to the reading of a nonconducting segment on the disk; then, transistors T_1 and T_2 conduct and transistor T_3 is cut off. In our example, the leading brush of the second bit, shorting the emitter circuit of transistor T_2, is located at a nonconducting segment. Consequently, transistor T_2 does not conduct. A large negative potential, denoting code 0, is at the output of this bit position (taken from the collectors of T_2 and T_3). If code 1 is in the first bit, the next bit must be read out from the trailing brush. Hence, transistor T_3 (connected to the trailing brush) must be conducting and transistor T_2 (connected with the leading brush) must be cut off. The presence of code 1 in the first bit position indicates the reading of a conducting segment of the disk. In this case, transistors T_1 and T_2 are cut off by the low negative potential at their bases, while transistor T_3 (connected to the collector of cutoff transistor T_1) conducts. If code 0 is in the second bit position, the trailing brush of this bit position is located on a nonconducting segment of the disk. Transistor T_3 is also cut off and a large negative potential (code 0) is obtained at the output of the second bit position. The above operating logic gives correct readings of all remaining bit positions of the code disk.

The above described shaft-position converter has six bit positions on the disk. For more accurate conversion of angular position, two b-bit code disks are connected by a reducing gear with a ratio of 1:64. This permits the reading of an angular position by 12 bits, (i.e., with an error not exceeding 1/4096).

The speed of a shaft with a code disk may reach 1500 rpm. Such converters may be used for digitization of angular coordinates.

Chapter VIII

Input-Output Devices

1. PUNCHED TAPE DEVICE

Input values given in tabular form may be directly inserted into the computer by means of punched tape or magnetic tape. Here, the use of punched tape as an input device is described.

Discrete increments $\Delta f(x)$ of the function $f(x)$ are recorded on paper tape in the form of perforations. Each pulse corresponds to a unit increment of the function and the sum of the accumulated increments (number of pulses) corresponds to the amount of the increments (number of pulses), i.e.,

$$f(x) = \sum \Delta f(x) + f(x)_0, \tag{1}$$

where $f(x)_0$ is the initial value of the function. In order to reduce the tape length, the derivative increments $\Delta f'(x)$ (i.e., second-order differences) are recorded instead of the increments of the function $f(x)$. Then, the accumulated derivative increments give the derivative of the given function, i.e.,

$$f'(x) = \sum \Delta f'(x) + f'(x)_0, \tag{2}$$

where $f'(x)_0$ is the initial value of the derivative.

Let us discuss in greater detail the insertion of a function of one variable $y = f(x)$ in the form of second-order differences. First, after numerical calculation of the quantity $\frac{\Delta f(x_i)}{h}$ at the points x_i $(i = 1,2,3, \ldots, n-1)$, a table of the second-order differences is prepared:

$$\Delta \frac{\Delta f(x_i)}{h} = \frac{\Delta^2 f(x_i)}{h}.$$

The x_i are assumed to be equally spaced (i.e., $x_{i+1} - x_i = h$ for all i). It is apparent that

$$f(x_{i+1}) = f(x_i) + \int_{x_i}^{x_{i+1}} \left[\frac{\Delta f(x_{i-1})}{h} + \Delta \frac{\Delta f(x_{i-1})}{h} \right] dx. \tag{3}$$

If it is assumed that in the segment between points x_i and x_{i+1} the function $f(x)$ may with sufficient accuracy be linearly interpolated, then from the last formula the approximate expression for $f(x)$ can be obtained for the present segment

$$f(x) \approx f(x_i) + \int_{x_i}^{x} \left[\frac{\Delta f(x_{i-1})}{h} + \Delta \frac{\Delta f(x_{i-1})}{h} \right] dx, \qquad (4)$$

where x lies between x_i and x_{i+1}. More accurate interpolation from formula (4) may be achieved for any continuous function by using a sufficiently large number of points x_i. In certain cases, expansion of the table for function $f(x)$ may be necessary.

Insertion of a function of one variable into the DDA is based on the use of formula (4). As the initial condition, the expression $\frac{\Delta f(x_i)}{h}$ is entered into the y register of the integrator and its increment is recorded on punched tape (i.e., $\Delta \frac{\Delta f(x_i)}{h}$ for $i = 1, 2, \ldots, n-2$). The punched tape is displaced by a distance which corresponds to a change in variable x by a value of Δx. A reader reads the increment $\Delta \frac{\Delta f(x_i)}{h}$ from the tape. This increment is either added to or subtracted from the code in the y register of the integrator for each change in x. The independent variable x is applied to the independent variable input of the integrator and the integrator thus interpolates the function $f(x)$ in accordance with formula (4). The increments of function $f(x)$ are obtained at the integrator output and may be used in the DDA. The use of formula (4) permits us to obtain the values of functions $f(x)$ without error in the interpolation unit if integration is performed correctly. The latter circumstance prevents the accumulation of error in interpolation of $f(x)$ over a large segment of values in x in precise integration of difference quotients $\frac{\Delta f(x_i)}{h}$.

The method for insertion of tabular or experimental values into the DDA is slightly different from that just described. The difference lies in the fact that the values recorded on the punched tape are not the increment quotients $\frac{\Delta f(x_i)}{h}$, but their increments, which are represented as +1 or -1 (the presence or absence of perforation) on the same scale for all increments. The transmission of inexact increments of $\Delta \frac{\Delta f(x_i)}{h}$ causes error in the values of $\Delta f(x_i)/h$ in the y register of the integrator. Even small errors in $\Delta f(x_i)/h$, in turn, may cause large errors at the integrator output if the range of values in x is sufficiently great. Hence the tape must be coded with consideration of eliminating the possible accumulation of error at the integrator output.

To insert a function of two variables, one may use the method which permits representation of a function of two variables $f(x, y)$ by functions of one variable, i.e.,

$$f(x, y) = f_0(x) + \varphi_0(y) + \sum_{i=1}^{\infty} f_i(x) \cdot \varphi_i(y). \tag{5}$$

A method for finding the functions f_0, f_1, $f_2 \ldots f_n$, φ_0, $\varphi_1 \ldots \varphi_n$ is described in ref. [7]. The method, in brief, is as follows. Let function $f(x, y)$ be given in the rectangle R: $a \leqslant x \leqslant b$; $c \leqslant y \leqslant d$. Moreover, let its mean value be

$$f_m = R \frac{\iint f(x, y)\, dxdy}{(b-a)(d-c)} = \frac{\iint\limits_R f(x, y)\, dxdy}{S_R}. \tag{6}$$

Examining the function $f^{(1)}(x, y) = f(x, y) - f_m$, we find the functions $f_0(x)$ and $\varphi_0(y)$ from the formulas:

$$f_0(x) = \frac{\int\limits_c^d f^{(1)}(x, y)\, dy}{d-c}; \quad \varphi_0(y) = \frac{\int\limits_a^b f^{(1)}(x, y)\, dx}{b-a}. \tag{7}$$

Let us further examine the functions $f_1(x)$ and $\varphi_1(y)$, which are determined from the equations:

$$f^{(2)}(x, y) = f^{(1)}(x, y) - f_0(x) - \varphi_0(y); \tag{8}$$

$$f_1(x) = \frac{\int\limits_c^d f^{(2)}(x, y) \cdot \varphi_{(1)}(y)\, dy}{\int\limits_c^d \varphi_1^{(2)}(y)\, dy}, \quad \varphi_1(y) = \frac{\int\limits_a^b f^{(2)}(x, y) \cdot f_1(x)\, dx}{\int\limits_a^b f_1^{(2)}(x)\, dx}. \tag{9}$$

Equations (9) are solved by the method of successive approximations. For the zeroth approximation for $f_1(x)$ and $\varphi_1(y)$, we may take arbitrary non-zero constants.

After the functions $f_1(x)$ and $\varphi_1(y)$ have been found, it is necessary to examine the function $f^{(3)}(x, y) = f^{(2)}(x, y) - f_1(x) \cdot \varphi_1(y)$. By means of this function, we find $f_2(x) \cdot \varphi_2(y)$ from the formulas:

$$f_{(2)}(x) = \frac{\int\limits_c^d f^{(3)}(x, y)\, \varphi_2(y)\, dy}{\int\limits_c^d \varphi_2^{(2)}(y)\, dy}; \quad \varphi_2(y) = \frac{\int\limits_a^b f^{(3)}(x, y)\, f_2(x)\, dx}{\int\limits_a^b f_2^{(2)}(x)\, dx}. \tag{10}$$

In the case of the function $f(x, y)$, all calculations from formulas (6)–(10) are performed numerically. The above method of representing a function of two variables by superposition of functions of

one variable permits replacing function $f(x, y)$ with a finite set of functions of one variable $f_i(x)\,\varphi_i(y)$ $(i = 1, 2, \ldots, n)$. This method is not suitable for insertion of $f(x, y)$ into the DDA if series (5) converges slowly. However, calculations have revealed that many functions may, with good accuracy, be represented in the form:

$$f(x, y) = f_0(x) + \varphi_0(y) + f_1(x) \cdot \varphi_1(y). \tag{11}$$

The recording of increments of a function of one variable on punched tape and the entry of the recording into the integrating computer is shown in Fig. 236a. The punched tape 1 is displaced in either direction in proportion to the value of x by means of a driver 2 with sprockets for transport of the tape. The tape is wound from reel 3 onto reel 4 or vice versa, depending on the direction of rotation. Increments of the functions are recorded on the tape by perforations. The number of holes punched along the tape is proportional to the value of the function, and the density of perforations (number of holes per cm length) is the derivative of this function. The maximum possible perforation density corresponds to the maximum value of the derivative of the given function.

The tape length is so chosen as to ensure the required accuracy in reproduction of the function. For example, if there are a total of 1000 perforations corresponding to change in the function from zero to maximum, then one perforation corresponds to 0.1% of its maximum value.

As the tape is transported, it passes under a hole-sensing contact head 5. When a hole passes under the head, contact 5 touches a metal plate beneath the tape and is thus connected to the positive side of the power supply. A voltage then appears at the input of gate 9.

There are three perforation tracks on the tape. On one track (or alternately on two tracks 6 and 7) there are uniformly spaced holes establishing tape displacement in divisions. These holes serve as "synchronizing" pulses, each being a marker of tape displacement by one division. With each displacement of the tape, the pulses read by heads 10 and 11 trigger flipflop 12. With each change of the flipflop state, a voltage pulse is produced and applied to gate 9. If a pulse from the flipflop coincides with a pulse read by head 5, the pulse passes through gate 9. This pulse represents the reading of a hole (code 1) on track 8 (on which the increments of the function are written) and the read-out code 1 is stored in flipflop 13.

The code stored in flipflop 13 is read out in the form of a pulse and transferred to the computer by applying a clearing pulse to the second input of flipflop 13 through gate 14. It is applied at the moment when the chosen integrator number arrives from the computer and the pulse is transferred to the chosen integrator. Gate 14 is normally conducting and is cut off only upon application of an inhibit pulse from flipflop 12 to its second input at the moment of tape displacement. This prevents possible loss of read-

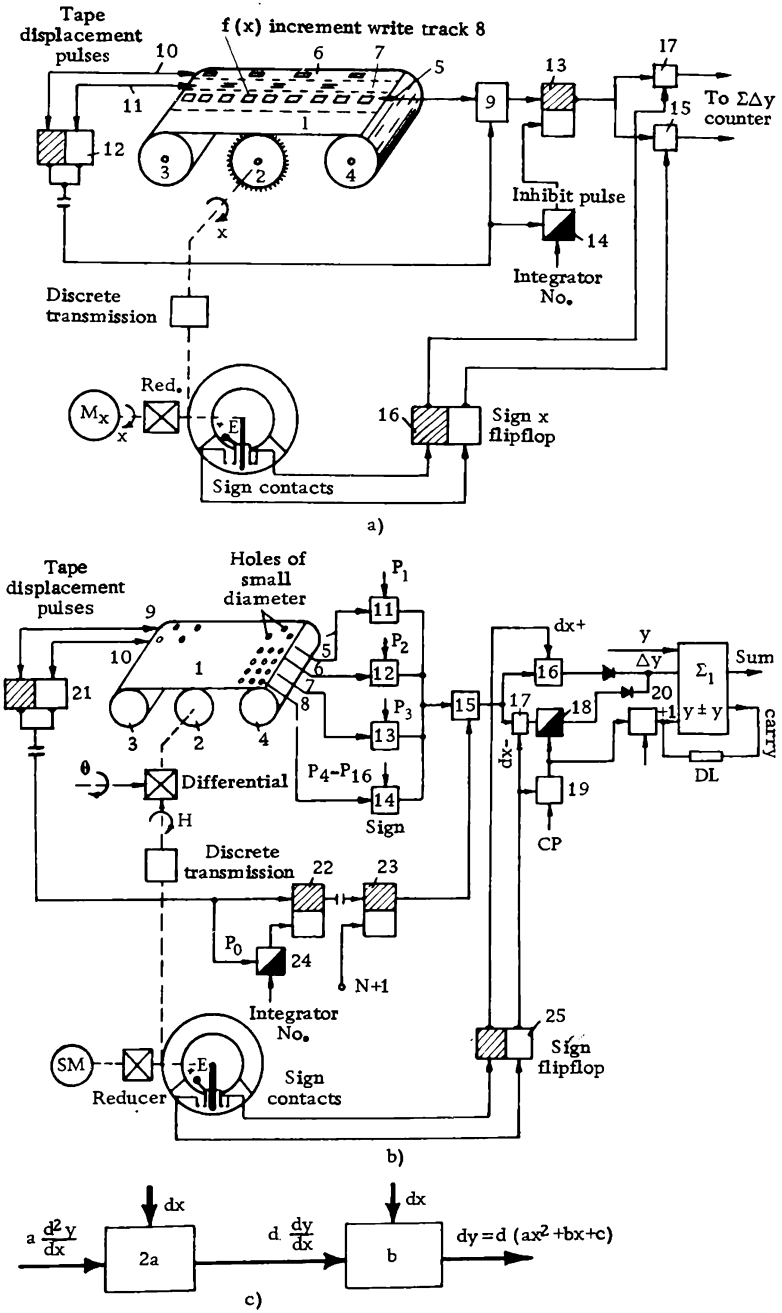


Fig. 236. Punched-tape function input: (a) entry of function onto punched tape; (b) entry of derivative increment code; (c) block diagram of quadratic interpolation.

out pulses written on the tape when the pulse from flipflop 12 coincides with the integrator number pulse, clearing flipflop 13. the tape is moved by motor M_x through a reducing gear Red. in proportion to the value of independent variable x. Displacement of the tape may be continuous or discrete. A contact device on the shaft of motor M_x (after the reducer) indicates the direction of shaft rotation x. It consists of an arm C on shaft x by which a contact carriage is frictionally displaced over the circumference of a disk. Depending on the direction of shaft rotation, one of the sign contacts is closed; these contacts control the state of sign flipflop 16. This flipflop, in turn, controls gate 15 or 17, which passes the pulses from flipflop 13 to the add or subtract input of the $\Sigma \Delta y$ counter. Thus, the sign of the increment of independent variable x is taken into account. The increments of the function $f(x)$ are assumed to have one sign for the entire range. If function $f(x)$ does not vary monotonically, a special track for the sign of increment $\Delta f(x)$ is used to reproduce the function $f(x)$. If the $\Delta f(x)$ increments written on the tape are used not only for the integrand of the integrators but also for the x input of the integrator, an intermediate servo integrator is used to deliver overflow pulses corresponding to the $\Delta f(x)$ increment written on the tape. If $\Delta f(x)$ has code 1 or 0, then code 1 or 0 is also obtained at the integrator output; but if increment $\Delta f(x)$ is equal to zero, a sequence of overflow pulses 1, 0, 1, 0, 1, 0, etc. is obtained at the integrator output as long as the increment of the function is zero.

Another method of recording data on tape is to perforate not the individual pulses, but the number codes of the increments of the function or the derivatives of the function.

It is known that a curve $y = f(x)$ can be approximated for a given accuracy by many line segments with the range of independent variable divided into as many equal intervals. For each line segment the derivative is constant since the slope of the line segment approximating the curve $y = f(x)$ is constant. If n values of the derivatives for n segments of the function, or, still better, derivative increments are recorded on the tape, these data will suffice to give the function $f(x)$ by integrating $f'(x)$ or $\int f'(x)\,dx$.

Each number code (an increment of the derivative) is recorded on the tape by punching a row of holes along the width of the tape (Fig. 236b). The number of parallel tracks is equal to the number of code bits, including the sign bit. Special tracks with read heads 9 and 10 are used to establish the tape displacement in divisions. The number code is read out in parallel by means of read heads 5-8. The parallel number code read out from the tape is converted into a serial pulse code by means of gates 11-14. These gates deliver the number code in a time sequence upon sequential arrival of timing pulses P_1, P_2, ..., P_n (corresponding to the first, second, third, etc., code positions).

The timing pulses come from the pulse-counter decoder in the integrating computer. If the number of bits of the recorded code is less than the total number of bits of the y register in the computer, then the codes of all the more significant bits are obtained in the

same manner as the sign code of the recorded number. This ensures correct summation of the tape-recorded derivative increment code with the derivative code stored in the integrator register.

Depending on the direction of tape displacement, the recorded code for increment $\Delta f'(x)$ must be added or subtracted from $f^1(x)$ in the y register. If the increment Δx is positive, the $\sum_1$ adder receives the direct code recorded on the tape through gate 16. If the Δx increment is negative, the $\sum_1$ adder receives the 1's complement code through gates 17 and 18. The 1's complement code is formed by anticoincidence gate 18, which passes the clock pulse CP to the adder input when the recorded bit has code 0 and inhibits the CP when the recorded bit has code 1. For correct subtraction, it is necessary to add 1's complements of the tape-recorded number plus 1 in the least significant bit of the addend. Addition of +1 to the adder is achieved through gate 20, passing the clock pulse for the first bit.

Selection of the tape codes at the required moment in computer operation is achieved by gate 15 once for each tape displacement by one division. Operation of flipflop 21, which controls the tape displacement, is the same as that of flipflop 12 in the previous arrangement. With each operation of flipflop 21, flipflop 22 is cleared by a pulse passing through gate 24 at the beginning of the integration interval when the tape code must arrive. Upon triggering flipflop 23, gate 15 conducts as long as addition of numbers occurs in the machine. This gate is cut off at the beginning of the next integrator number, when flipflop 23 is cleared. Flipflop 25 stores the sign of the increments and controls gates 16, 17 and 19 in the same manner as in the previous arrangement.

An advantage of recording the derivative increments rather than the derivative itself is that fewer bits and fewer read heads are needed.

Given a small number of codes required to record the derivative, a perforated cylinder made of an insulating material may be used instead of punched tape. Holes are punched on the surface of the cylinder according to the given code of the derivative. Within the cylinder of insulating material there is a metal cylinder which is connected to one side of the power supply. By discrete rotation of the cylinder, one or another number code of the derivative of $f'(x)$ is positioned under the contact heads. Approximately 40 14-bit binary numbers may be placed on a small cylinder having a diameter of 40 mm and a length of 50 mm. This amount of binary numbers for representing the derivatives is wholly adequate for generation of a function (for example, sine function) with an error of the order of hundredths of a percent.

Up to this point we have assumed that the derivative or increments of the derivative are recorded on the tape or cylinder, which corresponds to a linear approximation of the curve $y = f(x)$ by segments of straight lines. However, other methods of approximating a curve are possible. For example, in quadratic approximation, each segment of the curve is approximated by a

segment of a parabola, the equation of which may be written in the form

$$y = ax^2 + bx + c, \tag{12}$$

where a, b and c are constant coefficients for a segment.

By differentiating y with respect to x twice, we obtain,

$$\frac{dy}{dx} = 2ax + b; \quad \frac{d^2y}{dx^2} = 2a. \tag{13}$$

Hence it is seen that the second derivative for the given segment is a constant equal to $2a$, while the first derivative is a linear function of x.

By using two integrators (Fig. 236c), we may generate the function given by equation (12). In this method the integrand in the first integrator is $2a$; the initial integrand of the second integrator is b and the initial value of the integral is c. Then, by integrating with respect to the independent variable x, we obtain the function $f(x)$ in accordance with equation (12).

Quadratic approximation permits a considerable decrease in the number of segments for approximating the range of x. The number of segments depends on the required accuracy and the shape of the given function.

In recording a given function on punched tape or magnetic tape for use in the arrangement of Fig. 236c, it often suffices to record for each segment the values of the coefficients. Thus, from several reference points of the given curve $y = f(x)$, we may interpolate and obtain the point of curve y for any value of x. This process of interpolation reduces to an integration operation. For linear interpolation one integrator is required, for quadratic interpolation two integrators, and for cubic interpolation three integrators. The integrators of a digital integrating computer may be used for this purpose.

Quadratic or cubic interpolation provides high accuracy in reproducing a function with an extremely small number of approximated segments.

2. INPUT DEVICES FOR GRAPHICAL DATA

Graphical data can be inserted directly into an integrating computer by using a function generator as an input device. In this generator the function $y = f(x)$ is given in the form of a curve and is converted by means of a servo system into a mechanical angular displacement which is then sensed by a transducer.

The function generator (Fig. 237) consists of a drum 1 with a graph sheet wrapper on which the input function has been plotted in conductive ink. The angular position of the drum represents the argument x. A servo-driven curve follower 2, driven by servo motor M, follows the curve $y = f(x)$ on the paper. The curve

follower 2, an error detector, consists of two comb-like brushes 1 and 2 (Fig. 238) with fine nickel wires, insulated from one another on a nonconducting plate. A resistance layer 3 is applied to the surface of brush 1; the wires of brush 2 are shorted by a conducting plate 4. Brush 1 serves as a potentiometer and brush 2 as a current pickoff. Both brushes are in contact with the "conductive" curve $y = f(x)$ at one point 0. Upon deviation of the center of the follower from curve $y = f(x)$, a voltage ΔU proportional to the amount of the deviation is produced because the midpoint of potentiometer 3 is displaced relative to the curve, with which pickoff 2 is constantly in contact. Each brush is approximately 25 mm wide and contains approximately 100 nickel wires. The error voltage ΔU is amplified by amplifier 3 (Fig. 237), which controls the servomotor M. Continous tracking of the follower 2 along the curve is achieved by the servomotor. The shaft of motor M is coupled with the slider of potentiometer 4 from which a voltage proportional to $y = f(x)$ is obtained.

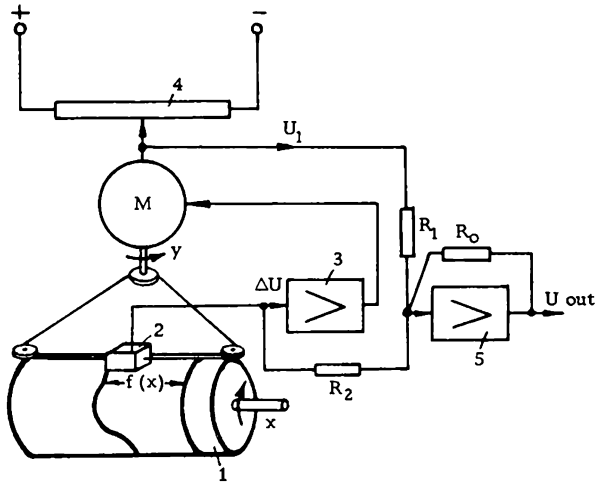


Fig. 237. A servo-driven function generator.

Dynamic error of the servo system in following the curve at high speed is reduced to a minimum due to the special compensation method described below. Operational amplifier 5 gives the voltage $U_1 + \Delta U$ by adding the currents passing through large resistors R_1 and R_2. Output voltage U_{out} of the operational amplifier correctly reproduces the given function even at extremely high tracking speed. Since amplifiers 3 and 5 have large gain, we have:

$$U_{out} = -\frac{R_0}{R_2} \cdot \Delta U + \frac{R_0}{R_1} \cdot U_1, \tag{14}$$

$$\Delta U = K_1 [f(x) - \bar{y}], \quad U_1 = K_2 \cdot U, \tag{15}$$

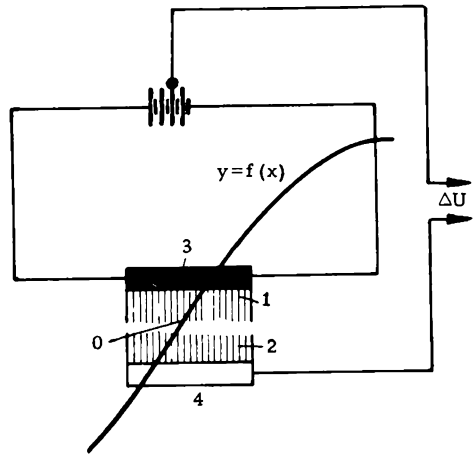


Fig. 238. Curve follower.

where $f(x)$ is the given function on the plot (the ordinate);

$\bar{y}$ is the displacement of the follower;

R_0 is the feedback resistance of operational amplifier 5;

K_1 is the coefficient of proportionality between the deviation of the follower 2 and voltage ΔU;

K_3 is the coefficient of proportionality between displacement $\bar{y}$ and voltage U_1.

From equations (14) and (15) we obtain

$$U_{out} = -\frac{R_0}{R_1} K_3 \bar{y} - \frac{R_0}{R_2} K_1 \cdot f(x) + \frac{R_0}{R_2} K_1 \cdot \bar{y}. \qquad (16)$$

Parameters R_1, R_2, R_0, K_1 and K_3 may be so chosen that the first and last terms in equation (16) cancel out. In this case, voltage U_{out} (over a range of deviation not exceeding one half the size of the follower) is proportional to $f(x)$ and tracking error is eliminated. Experimental study of such a system has shown that even a step function was correctly reproduced in a total tracking time of several tenths of a second. The tracking speed along the drum circumference (X axis) may reach 250 mm/sec and along the drum length (Y axis) 170 mm/sec; the graph paper is 2.2 by 2.75 m. Static error of the function generator is 0.2%. This arrangement may also be used to record a voltage which is a function of another voltage. In this case, the follower 2 is replaced with a writing pen. The above arrangement has the following advantages: simplicity, high accuracy and speed in inserting a function, possible compensation of the dynamic error of the servo system, and, the possibility of its use for generation of various functions at low frequency.

The function generator in Fig. 237 is used in the setup in Fig. 239 to obtain increment pulses of the given function $y = f(x)$ and to enter them into the computer.

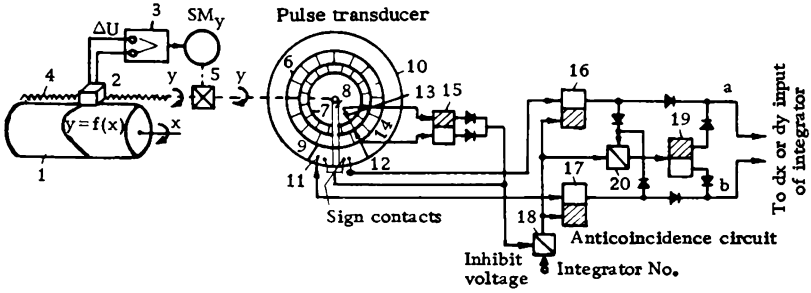


Fig. 239. Reading of increment pulses from a function generator.

A pulse transducer coupled with the shaft of servomotor SM_y through reducer gear 5 converts the angular displacement of the shaft to a proportional number of pulses. This is achieved by means of two contact discs 6 and 7; the contact segments have a relative displacement of one half contact division.

Contact arm 8, which moves contact carriage 9, is rigidly coupled with shaft y; the latter is held against disk 10 by spring pressure and moves over its surface in guide grooves. Upon movement of the carriage, contact arm 8 closes the left contact 11 or right contact 12; these two contacts determine the direction of shaft rotation. The size of the gap between contacts 11 and 12 is determined by the permissible dead space for angular rotation of the shaft. Contact brushes 13 and 14 are mechanically controlled by carriage 9 and slide over the two contact disks 6 and 7 with insulated contact plates.

All contact plates are connected to one side of the power supply. Upon movement of brushes 13 and 14 during shorting of the brushes with the contact plates, voltage pulses are alternately applied to the two inputs of flipflop 15. The flipflop is triggered by the pulses and generates rectangular pulses (positive or negative) stored in flipflop 16 or 17, depending on the direction of shaft rotation.

With return of flipflops 16 and 17 to the initial state (clear), a pulse code corresponding to the increment of shaft y ($+1$ or -1) is applied to input Δx or Δy of the required integrator. Flipflops 16 and 17 are cleared through gate 18 (coincidence circuit) by application of a pulse from the required integrator line at the appropriate time.

In order to obtain a sequence of pulse codes 1, 0, 1, 0, 1, 0, etc., an additional flipflop 19 and an anticoincidence circuit 20 are used. If there are no increment pulses from shaft y, flipflop 19 is triggered with each integration step and delivers alternately to lines a and b pulses corresponding to code 1, 0, 1, 0, 1, 0, etc., at the integrator input. In the presence of an increment pulse from shaft y, anticoincidence circuit 20 inhibits the pulse to flipflop 19. Thus, the shaft increments are converted to pulses which are then inserted into the required integrator as function $f(x)$.

By means of a relatively simple arrangement, any function $f(x)$ given in graphical form may be recorded in coded form on punched tape, which is then used for insertion of the given function into the computer. This permits the use of only one function generator for many of the functions which must be inserted into the computer. Figure 240 shows the diagram of an arrangement for punched-tape coding of a function given in graphical form. The function generator consists of drum 1, follower 2, amplifier 3 and servomotor SM_y.

Punched tape 6 is displaced synchronously with drum 1 in accordance with independent variable x. The punched tape is displaced in discrete steps (one division at a time) by means of a discrete drive 7. The latter is the same as that used in an electromechanical counter. A finite angular increment of shaft x causes tape displacement by one division. Perforation of the tape for independent variable x is achieved as follows. With each division of shaft displacement Δx, an electromechanical pulse transducer 8 delivers a pulse which is stored in flipflop 9. Electromagnet 10 (x-track perforator) is then actuated, punching a hole on track x of the tape. During the punch operation, contacts 11 of the electromagnet are closed (at the end of the magnet travel) and a voltage pulse clears flipflop 9 with the result that the electromagnet returns to its initial position.

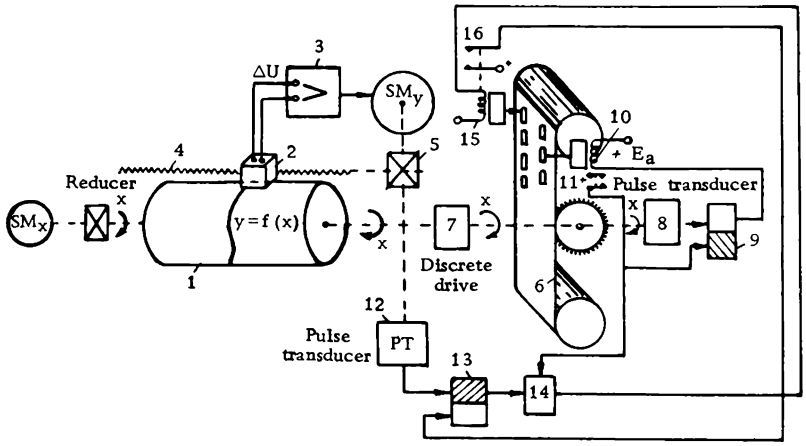


Fig. 240. Conversion of graphical data into punched-tape codes.

If there occurs an increment of function y in a shaft displacement Δx, pulse transducer 12 delivers a pulse which is stored in flipflop 13. Flipflop 13 controls gate 14, which (in the presence of increment Δy) passes a voltage pulse from contacts 11 to electromagnet 15 of the y-track perforator. In this manner each perforation of the y track is achieved after each perforation of the x track. Both the x and y perforations lie on a row along the tape

width. After perforation on the y track, flipflop 13 is returned to the initial state by a pulse from the closure of contact 16 of electromagnet 15 (at the end of the electromagnet punching stroke).

Tape displacement and curve following must be slow enough to allow sufficient time for perforation of the tape. Read-out of the punched-tape data into the computer can be achieved somewhat more rapidly if second-order differences (increments of the derivative of the function) are recorded on the tape.

3. GRAPHICAL PLOTTING OF COMPUTED RESULTS

Computed results can be recorded in curve form by means of a plotter (Fig. 241). The increments Δy which are accumulated during each integration step are transferred from the ΔS line to the $\sum \Delta y$ counter and then to the y register. The same counter and register also receive the Δy increment pulses from the plotter as feedback. Thus, the y register of the integrator contains a number which is equal to the difference

$$y - \overline{y} = \Delta,$$

where y is the computed quantity and $\overline{y}$ is the quantity generated at the y shaft of the plotter.

The difference Δ is converted into an electrical voltage which controls servomotor SM_y to generate the computed quantity $\overline{y}$ in the form of a mechanical displacement of the y shaft of the plotter. The plotter consists of a drum 1 on the surface of which function $y = f(x)$ is automatically plotted by the write head 2. Write head 2 is moved along a lead screw 3 driven by the servomotor through a gear reducer 4.

An electromechanical pulse transducer similar to that shown in Figure 239 controls displacement of the write head via the gear reducer. Pulses corresponding to a certain increment $\overline{\Delta y}$ are applied as a negative feedback to the input of the $\sum \Delta y$ counter of the integrator and then become increments to the computed quantity Δy. As a result, the y register contains the magnitude of the difference between the computed and the generated values of y. The code for the difference Δ is a sequence of pulses through gate 5, which is controlled by a given integrator number. This code is stored sequentially in flipflops 10-12 and sign flipflop 13 through gates 6-9 by pulses P_1, P_2 ... P_n from the decoder of the digit counter. Gate 8 is unblocked by any pulse from P_3 to P_{n-1}, and gate 9, which passes the sign code (see Chapter V, Section 7), is unblocked by last pulse P_n.

It is not necessary to have as many flipflops as bit positions in the code for the difference Δ in order to control the motor. For example, it suffices to store the first two or three bits of the Δ code in flipflops 10-11, while code 1 in any of the more significant bit positions of the Δ code can be stored in flipflop 12. The high voltage level of this flipflop corresponds to maximum speed of the motor SM_y.

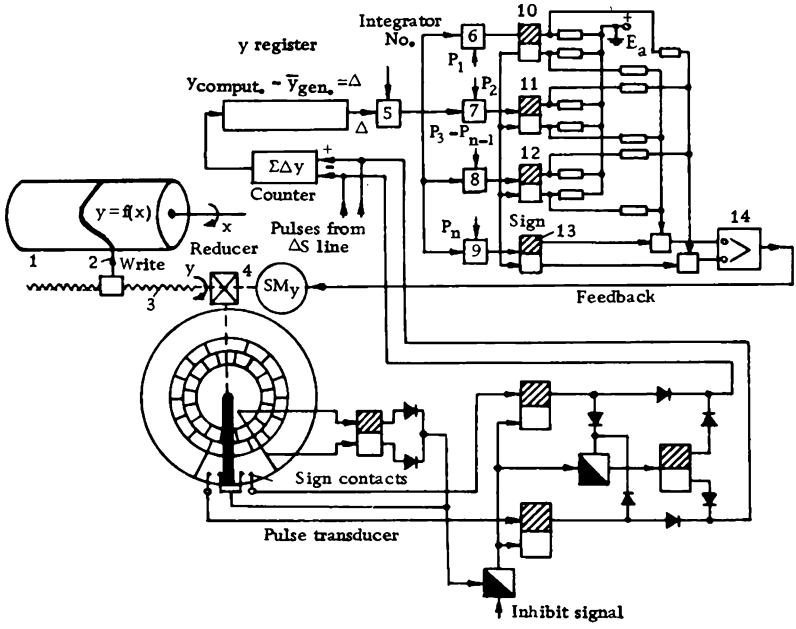


Fig. 241. Graphical plotter for recording computed results.

In fact, the difference $\Delta = y - \bar{y}$ should be small in the control process, and three to four bits are sufficient for smooth control of motor operation.

The state of the flipflops containing the Δ code is converted into an electrical voltage which controls the servomotor SM_y. This conversion is achieved by a resistor network and summing amplifier 14. The curve $y = f(x)$ plotted on the drum accurately reproduces the computed values of y, since each increment Δy of the computed y is generated by SM_y in the form of a corresponding mechanical displacement as a result of the feedback loop.

4. HIGH-SPEED FUNCTION GENERATORS

Many experimentally derived functions are in tabular or graphical form which cannot be represented by a simple, analytical expression over an entire range of the independent variable. Such functions can be conveniently inserted into the digital computer by means of special function generators.

The better known methods for insertion of functions by means of a punched tape or a magnetic tape possess the disadvantage that direct data read-out from the punched tape or magnetic tape limits the computing speed of the machine. In addition, the electromechanical tape transport and the method of reading out the

recorded data complicates the input unit, particularly when it is necessary to insert numerous functions into the computer.

The methods for high-speed (of the order of 1Mc) insertion of tabular and experimental data into a digital integrating computer, as described below, are proposed by the author.

The function generator (Fig. 242) generates increments of the first or second (or even higher) derivative of a given function $y = f(x)$ in the form of individual pulses. These pulses are stored in the y register of an integrator by which intermediate values of the given function are interpolated. For example, if the function generator delivers increment pulses for the first derivative (second-order differences), the increments of function $y = f(x)$ are computed by means of a single integrator:

$$\int dy = \int f'(x)\,dx.$$

If the function generator delivers an increment of the second derivative (third-order difference), computation of the increments of the function is achieved by two integrators, etc. The increment pulses for any derivative of the given function are stored, for example, in 16 ferrite-core transformers, each of which delivers at its output winding a pulse if the increment code is 1 and no pulse if the increment code is 0. In the latter case, the corresponding core is not threaded through.

For each value of the independent number code x there is a corresponding increment of the function at the output of the function generator. For example, if the number code x is given in the form of a 10-bit binary number, the function generator gives 1024 ones as the increment of the function. These values are obtained by threading 64 lines (or four groups of 16 lines) through 16 ferrite-core transformers. By means of diode decoders and a diode-transformer decoder, the four least significant bits of the x number code control 16 output gates $1'$-$16'$ to deliver the function increment code stored in the 16 ferrite-core transformers. The 16 cores are threaded with 64 lines arranged in four 16-line groups. The 16 lines correspond to the 16 possible values of code x for the four bits of x having the weights 2^4, 2^5, 2^6 and 2^7, and the four groups correspond to the four possible values of code x for the two bits having the weight 2^8 and 2^9. By means of a diode decoder, the latter two bits control four group gates B_1–B_4 which connect the chosen group of 16 lines to ground. Pulses to the 16 lines of each group are provided by a diode and diode-transformer matrix connected through diode decoders to the four bits of number x having the weights 2^4, 2^5, 2^6 and 2^7. Thus, the function generator produces increment code pulses for 1024 discrete values of argument x (i.e., 1024 values of the given functional curve). The intermediate values of the function are interpolated by integration.

The sign of the increment is determined by gates B_a and B_b, which are actuated by increment pulses Δx_+ and Δx_- corresponding to a positive or negative increment of the argument.

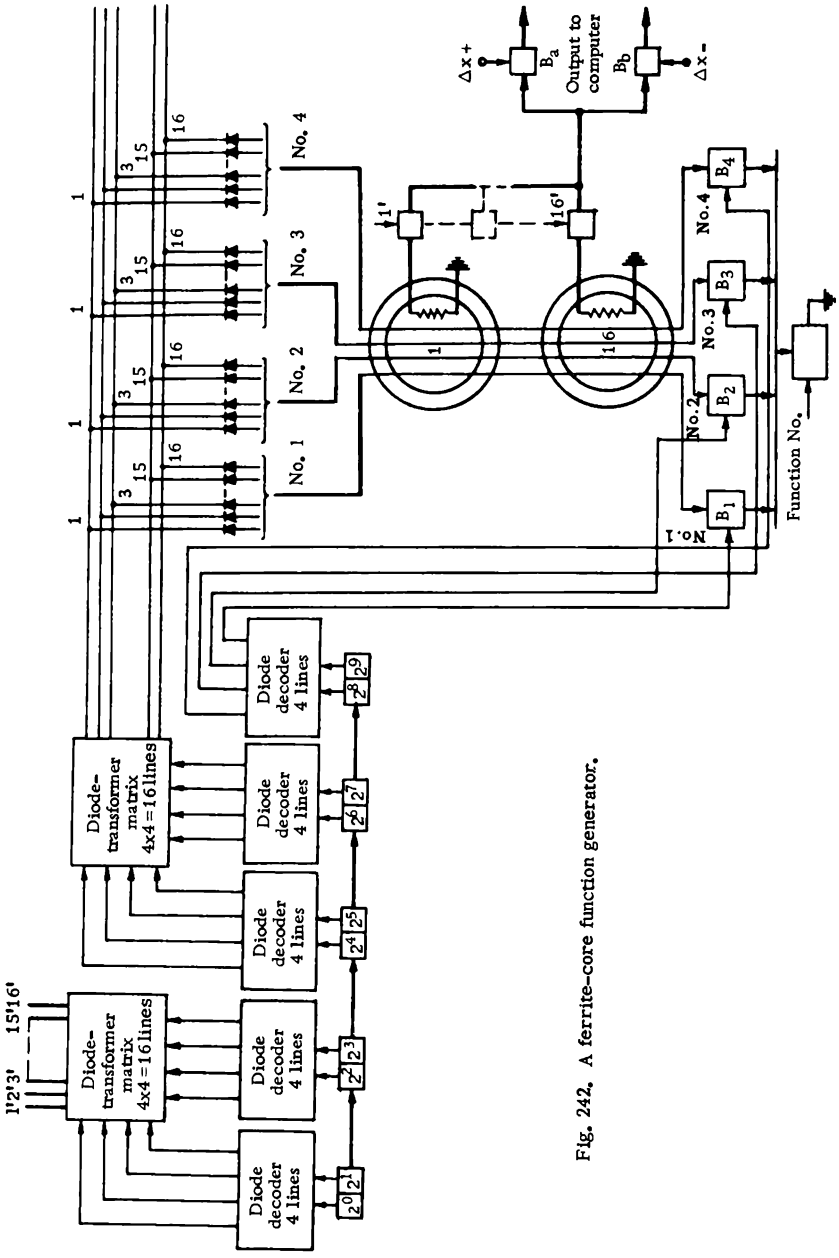


Fig. 242. A ferrite-core function generator.

The number x decoders thus achieve "electronic scanning" of the increment data stored by threading of the transformers in a manner similar to that in which the transport of a punched tape achieves mechanical scanning. The above function generator is used for insertion of several functions. For this purpose, the code of number x is inserted into the decoder register from the memory and the threaded cores corresponding to various functions are switched by a function number gate.

Another method of inserting functions into a computer is to insert the number codes of the derivatives or of the coefficients of a polynomial approximating different segments of the curve. For example, in approximating the curve $y = f(x)$ by segments of a parabola $y = ax^2 + bx + c$, the coefficients a and b are given by a function generator (Fig. 243). Computation of the dy increments for any continuously varying value of argument x is achieved by means of two integrators. The n-bit codes of numbers a and b are obtained for entry into the computer from ferrite-core transformers $1, 2, 3 \ldots n$ and $1_1, 2_1, 3_1 \ldots n_1$ (Fig. 243). The codes of numbers a and b are stored by means of the primary windings of the transformers, wound for code 0 and not wound for code 1. The series-connected primary windings of the transformers (for all bits of the number code) are pulsed by the lines from a diode-transformer matrix; each line corresponds to a discrete change of argument x. If it is necessary to insert into the computer the number codes of a and b for 10 values of argument x, 10 output lines of the diode-transformer matrix, corresponding to the 10 values of argument x, are selected.

The output lines of the diode-transformer matrix are selected by two ring counters connected in series. These counters are capable of reversible counting of the Δx increment pulses within the given range of x. The codes for numbers a and b are entered into the computer serially (at a frequency of 1-2 Mc) by means of gates B_1—B_n, which are controlled by bit pulses P_1—P_n of the integrating computer. These number codes arrive at the y register of the chosen integrator through gates B_y, actuated by the number line of the chosen integrator of the computer.

The number codes a and b, stored in two rows of cores $1, 2, 3, \ldots, n$ and $1_1, 2_1, 3_1, \ldots n_1$, may be wound on one row of cores $1, 2, 3, \ldots, n$. This permits a 50% reduction in the number of gates $B_1, B_2, \ldots, B_n$ and simplification of the other circuits.

For quadratic or cubic interpolation of the function by means of a polynomial, the coefficients a and b may also be stored in the magnetic internal memory (MIM) of the computer. In this case, a diode-transformer matrix (Fig. 244) is used as a decoder and the output lines of the matrix are selected in accordance with the given argument of x. The selected output line of the diode-transformer matrix connects the number line of the MIM along which the coefficients $a_1, a_2 \ldots a_n$ are stored in the cores.

A single diode-transformer matrix may be used for selection of many functions of different arguments x. The output lines of this matrix are selected according to the number code of the

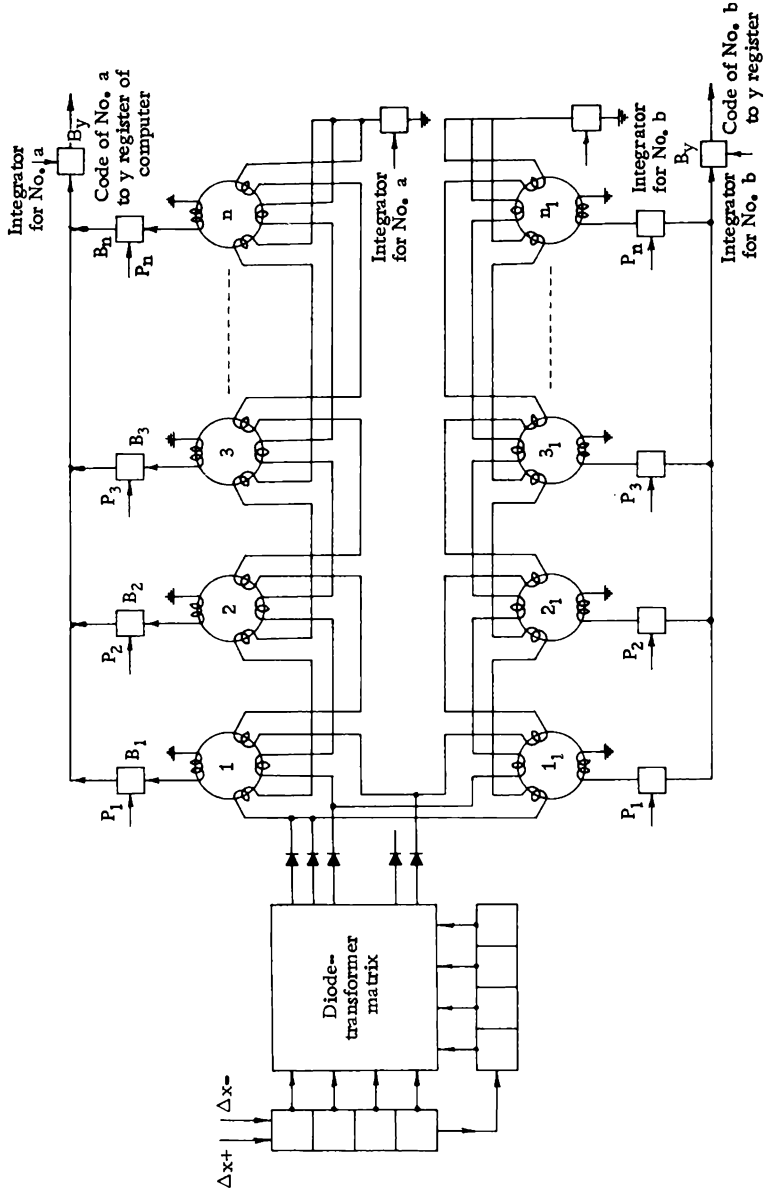


Fig. 243. Generator of polynomial coefficients approximating segments of a curve.

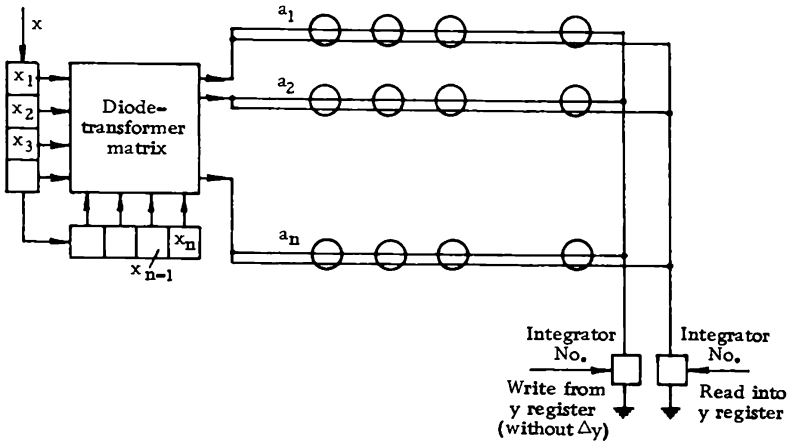


Fig. 244. A magnetic-core memory for storing polynomial coefficients for quadratic or cubic interpolation.

argument, which is entered into the register controlling the matrix. For several sets of coefficients a_1, a_2, a_3... a_n for several functions, several read-write lines must be provided. These lines are chosen by gates and controlled by the lines of the chosen integrator numbers. If a permanent ferrite-core memory is used for storing the coefficients of the several functions, there must be several sets of windings on the cores.

For certain practical applications it is necessary to obtain a certain functional dependence $y = f(x)$ in the form of the code of number y for a given code of number x. Let us discuss a generator (Fig. 245) in which the argument x is given in the form of pulse-time code arriving over a telemetry channel which transmits several input quantities. For each input quantity it is necessary to obtain the functional dependence.

The initial pulse of each code sets flipflop FF_1 to the 1 state, which unblocks gate B_I to pass CP pulses; these pulses are then counted for argument x by the pulse-time code counter. Gate B_I is cut off by the final pulse of the pulse-time code, which resets FF_1 to the 0 state. The number of CP pulses passed through gate B_1 and counted by the counter during the time between the initial and final pulses of the pulse-time code corresponds to the input variable x. The counter, consisting of two series-connected ring counters, controls the diode-transformer matrix. The output lines of this matrix select the windings of ferrite-core transformers 1_1—2_1 ... 30_1, from which the code of the function $y = f(x)$ is generated. For each input variable (each channel) the functional code is stored in one row of ferrite-core transformers, the windings of which are selected on one end by the diode-transformer matrix and on the other end by gates B_1, B_2, ..., B_{10}; this selection is controlled by pulses from the output lines of the input channel counter. The secondary windings of each corresponding bit of the

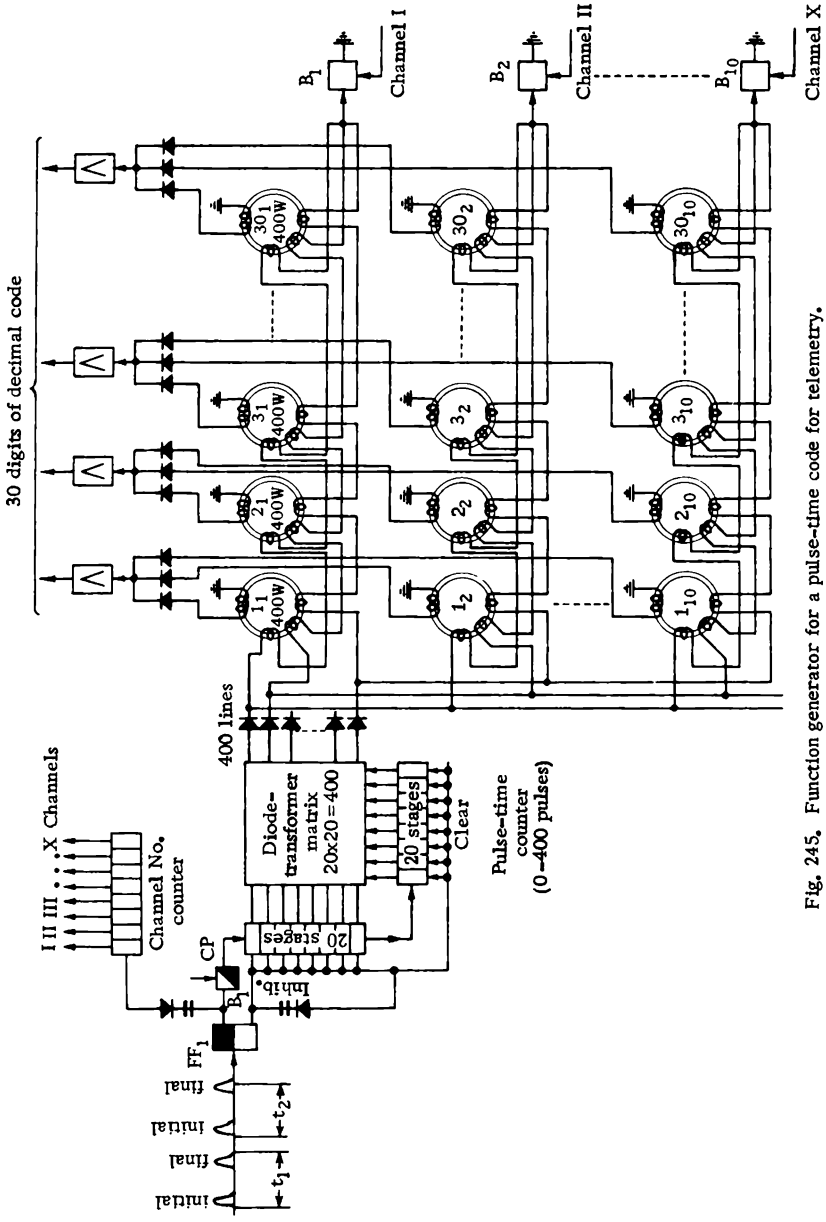


Fig. 245. Function generator for a pulse-time code for telemetry.

functional code are connected in parallel for all channels. The pulse-time code is converted to the functional code in the decimal number system with an accuracy of 0.25%. For 10 input channels the generator provides 10 functional dependences. The diode-transformer matrix has 400 switchable lines for 400 values of argument x. The entire generator uses 110 transistors, 1260 diodes, and 880 ferrite-core transformers.

The above discussed methods for generating functions of one variable may also be used for generating functions of two variables. As was shown in Section 1 of this chapter, a function of two independent variables

$$U = f(x, y) \tag{17}$$

may be approximated by a formula containing four functions of one variable:

$$U = f(x, y) = f_1(x) + f_2(y) + f_3(x) \cdot f_4(y). \tag{18}$$

Equation (18) in differential form becomes

$$dU = df(x, y) = df_1(x) + df_2(y) + f_4(y)\, df_3(x) + f_3(x)\, df_4(y). \tag{19}$$

Each of the functions of one variable may be computed in increments of its first derivative and stored for each increment dx, i.e.,

$$df(x) = df'(x)\, dx.$$

Computation of a function of two variables (Fig. 246a) requires seven integrators: four for the generation of derivative increments of the four functions (second-order differences), two for multiplication, and one for summation of the four terms of equation (19). The schematic shown in Fig. 246b is used for generating the derivative increments of the function in integrators 1 and 2.

In this figure the code pulses $df_1'(x)$ and $df_3'(x)$ are produced at the output windings of transformers TR_1 and TR_2. The presence of pulses (with CP frequency) corresponds to code 1 and the absence of pulses corresponds to code 0. The required sequence of codes for different discrete values of x are stored by the primary windings of the transformers, which are a "permanent storage" for the code pulses. If the entire range of independent variable x is divided into 100 equal parts (i.e., if there are 100 discrete values of x), then there are 100 windings in each of the transformers. If the code pulse for a derivative increment of the given function is 1, then the transformer winding for the given value of x is connected to the current loop; if this location contains code 0, the winding is bypassed from the current loop.

The series-connected windings of transformers TR_1 and TR_2 are driven by a diode-transformer matrix the 100 windings of which are connected respectively to the 100 windings of transformers

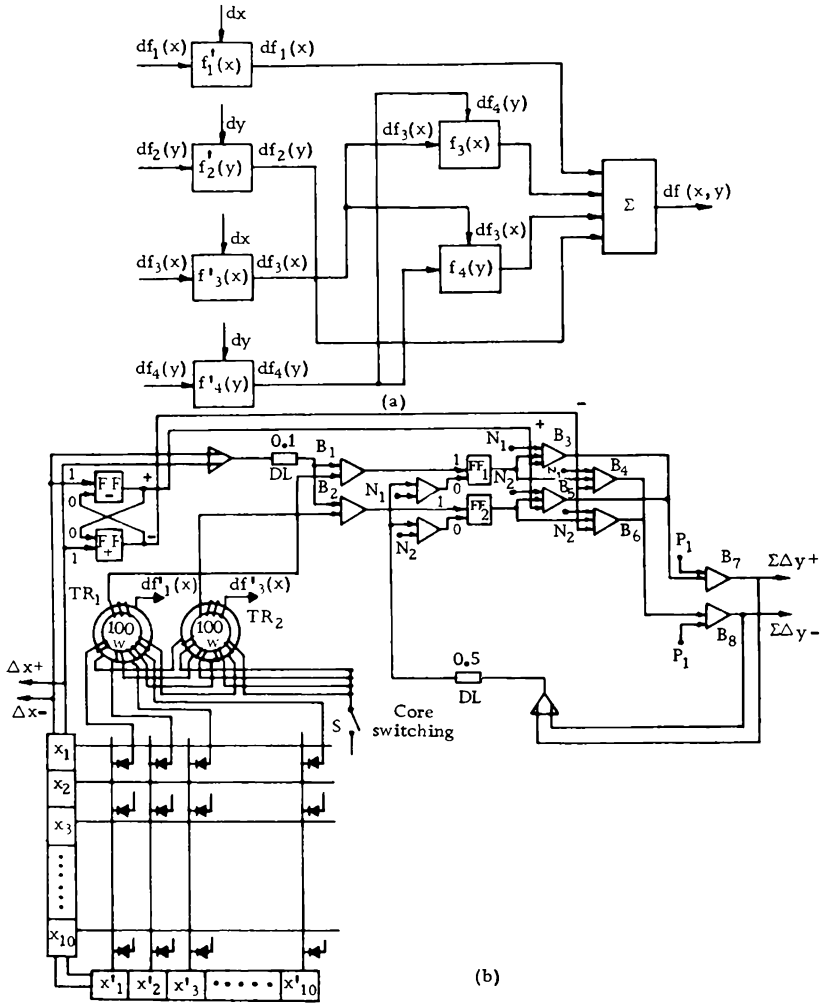


Fig. 246. A two-variable function generator. (a) block diagram; (b) schematic diagram.

TR_1 and TR_2. The number of transformers may be increased. In this case the number of diode-transformer matrix lines is correspondingly reduced. The matrix is composed of ten horizontal and ten vertical lines and is controlled by two series-connected ring counters for counting up to 100 Δx increment pulses (positive or negative). Each counter consists of ten stages (dynamic flipflops). For each value of x, one winding of the matrix is selected and this winding delivers the corresponding code for the derivative increment. The sign of the increment of independent variable x is stored in dynamic flipflops FF_+ and FF_-, which partially control the gates for delivering the code

pulses to the computer. The code pulses of the derivative increments are inserted for each increment in independent variable x by gates B_1 and B_2. In this case, dynamic flipflops FF_1 and FF_2 store the code pulses, which are delivered through gates B_3 or B_4 (depending on the sign of increment Δx) at the appropriate moment to the add or subtract input of the $\sum \Delta y$ counter of the integrating unit (depending on the pulse code being 1 or 0). Gates B_3 and B_4, and B_5 and B_6 are unblocked by the number lines of those integrators to whose Δy input the derivative increments of the function are transferred. Gates B_7 and B_8, which pass to the counter only one code pulse, are controlled by the first bit pulse P_1. After passage of the code pulse to the counter, the appropriate dynamic flipflop (FF_1 or FF_2) is cleared by a pulse from gates B_7 and B_8 after a 0.5 μsec delay.

The above described generator may be used for generating several functions of one independent variable by sequential switching of several transformers. If it is necessary to switch from one function to another, switch S may be used to switch the windings of the various transformers; an identical arrangement is used for generating the derivative increments of functions of a second independent variable y. Switch S may also be expanded for use in generating a function of three independent variables.

Chapter IX

A High-Speed Parallel Digital Integrating Computer

1. BLOCK DIAGRAM OF A PARALLEL INTEGRATING COMPUTER

The parallel digital integrating computer, designed for analog computation and real-time control, consists of units (Fig. 247) which achieve the following functions:

a) Initial data entry and multichannel conversion thereof from analog to digital form;
b) Integration;
c) Insertion of initial data into integrators;
d) Interconnection of integrators;
e) Data output;
f) Pulse generation.

The unit for data input from external channels receives input variable increments for each integrator. If the problem solution time is one second, during which the computer performs 100,000 iterations, and if input data accuracy is 0.01%, it is necessary to receive 10,000 increments per second for each integrator. A multichannel electronic converter can perform 100,000-200,000 input data conversions per second and is thus able to provide input data conversions for 10-20 channels which are switched in the input unit at a frequency of 100-200 kc. The converter output is directly connected to one or more integrators.

Input functions are entered into the parallel computer by a ferrite-core function generator (see Chapter VIII, Section 4) which may attain an input increment frequency of 100 kc.

The integrating unit consists of integrators, multiplier integrators, servo (comparator) devices, function generators, and increment adders.

The program unit establishes interconnection of the above mentioned devices in the integrating unit in accordance with the program to be solved. Interconnection is achieved, for example, by means of plug-in cables.

The initial data unit consists of a memory for storage of the initial values of the integrands. These values are serially inserted into the y registers of the integrators.

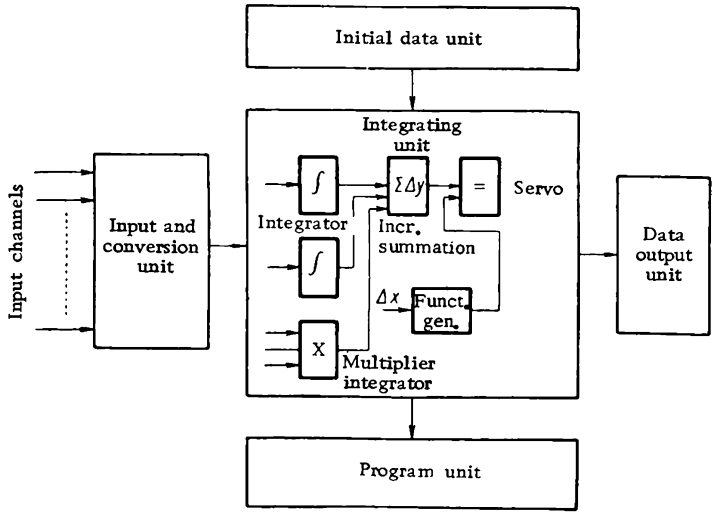


Fig. 247. Block diagram of a parallel integrating computer.

With a relatively large number of integrators it is advisable that the initial-condition storage be common to all integrators of the computer. The use of a common memory facilitates connection of a parallel DDA with an arithmetic machine which computes the initial data for the DDA and also permits insertion of data from other external channels.

The registers for storing the initial values may be included in each integrator, for example, in the form of a delay line (Fig. 248) as a "dynamic memory." However, this complicates the integrator, because an additional read-write amplifier as well as flipflops and gates are required for control of the initial-value register. Such an arrangement for storage of the initial data in each of a large number of integrators requires considerably more electronic components than the use of a common memory.

The data output unit is similar to that in the serial integrating computer (see Chapter V, Section 7). The error signals between the computed and "generated" values of the controlled variable reach the output in parallel but are converted into a serial code.

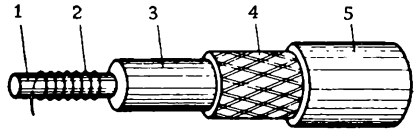


Fig. 248. Coaxial cable as a delay line: 1—insulating (or ferrite) core; 2—helical conductor; 3 and 5—insulating layer; 4—metal braid.

The controlled variable may be actuated by a servomotor or by an electronic device (electronic counter, register). With two parallel counters at the output, the binary code continuously generated at the output may be converted to decimal code. Such conversion is of practical advantage for a small number of output channels.

The pulse generator contains a number of high-power CP generators (at a frequency of one or several Mc) using transistor circuits. These generators produce pulses from a common master clock.

2. DELAY-LINE STORAGE REGISTER

The coaxial-cable delay line with distributed parameters L and C is a dynamic memory and can be used as the register in an integrator of a parallel integrating computer.

A high-frequency coaxial delay line (Fig. 248) with distributed parameters consists of an internal spiral conductor 2 wound in a single layer on an insulating (or ferrite) core 1. Such a structure results in an increase in the inductance per unit length of cable and thus gives an increase in delay time.

The delay time t_{del} of a cable is

$$t_{del} = t_{dl} \cdot l = \sqrt{L_0 C_0} \cdot l, \tag{1}$$

where L_0 and C_0 are the inductance and capacitance per unit length; t_{dl} is the delay per unit length of cable; l is the length.

The characteristic impedance of the cable is

$$\varrho = \sqrt{\frac{L_0}{C_0}}. \tag{2}$$

Characteristics of some cable delay lines are:

RKZ-401

Characteristic impedance, ohms	350-500 (+10%)
Length of section, m	10
Delay time (larger than or equal to), μsec/m	0.5
Attenuation (less than or equal to), nep/μsec at 10 Mc	0.55
Cable weight, kg/m	0.135
Permissible bend in spiral, mm	100
Temperature range, °C	40 to 75

RKZ-1601

Characteristic impedance, ohms	1600
Delay time, μsec/m	1.6
Attenuation, nep/μsec at 2 Mc	0.12

Available literature on a cable section used as a delay line for a digital integrator gives the follow data:

Delay time of cable section, μsec	9.33
Characteristic impedance, ohms	13.50
Maximum pulse repetition frequency, Mc	3
Number of bits stored in the line	28
Pulse rise time, μsec	9.25
Attenuation, db	6

In order to decrease the requirement for the bandwidth of the cable, code pulses in the cable are represented by two voltage levels without spacing between the codes (i.e., so-called "nonreturn-to-zero" method (Fig. 249)). In this method, the highest frequency required of the delay line is twice the frequency obtained in the ordinary pulse method with intervals between the codes. The "nonreturn-to-zero" method requires read-write amplifiers with direct coupling between stages.

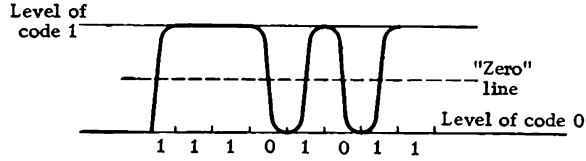


Fig. 249. Nonreturn to zero method of binary representation.

The schematic for a cable delay line is shown in Fig. 250. The delay line consists of:

a) a write flipflop;
b) a section of cable delay line;
c) a read amplifier;
d) a read flipflop.

The write flipflop, a static flipflop, has two inputs for entry of code 1 and code 0 by negative pulses. The flipflop proper uses two p-n-p transistors (type P401) T_1 and T_2. The two other transistors T_3 and T_4 are input amplifiers. The flipflop is triggered by CP pulses from the clock generator at a frequency of 3 Mc and with a pulse duration of 0.15 μsec. Upon coincidence of the input signal with the CP pulse, diode D_1 is cut off and a negative pulse arrives at the base of transistor T_3 or T_4. In order to inhibit a trigger pulse when the flipflop is in the required state, the flipflop outputs are cross-coupled. The flipflop output voltages cut off diodes D_2 and D_3 if the voltage at the flipflop output is negative (T_2 cutoff and T_1 conducting), and only in this event does a trigger signal become effective. Capacitor C at the transistor amplifier differentiates the trigger pulse and isolates transistors T_1 and T_3. Diode D clamps the backswing of the differentiated trigger pulse. Pulse rise time at the output of transistors T_1 and T_2 is approximately 0.1 μsec.

The cable delay line constitutes a circulating register for storing a code of 28 pulses (or bits) with a duration of 1/3 μsec. Thus,

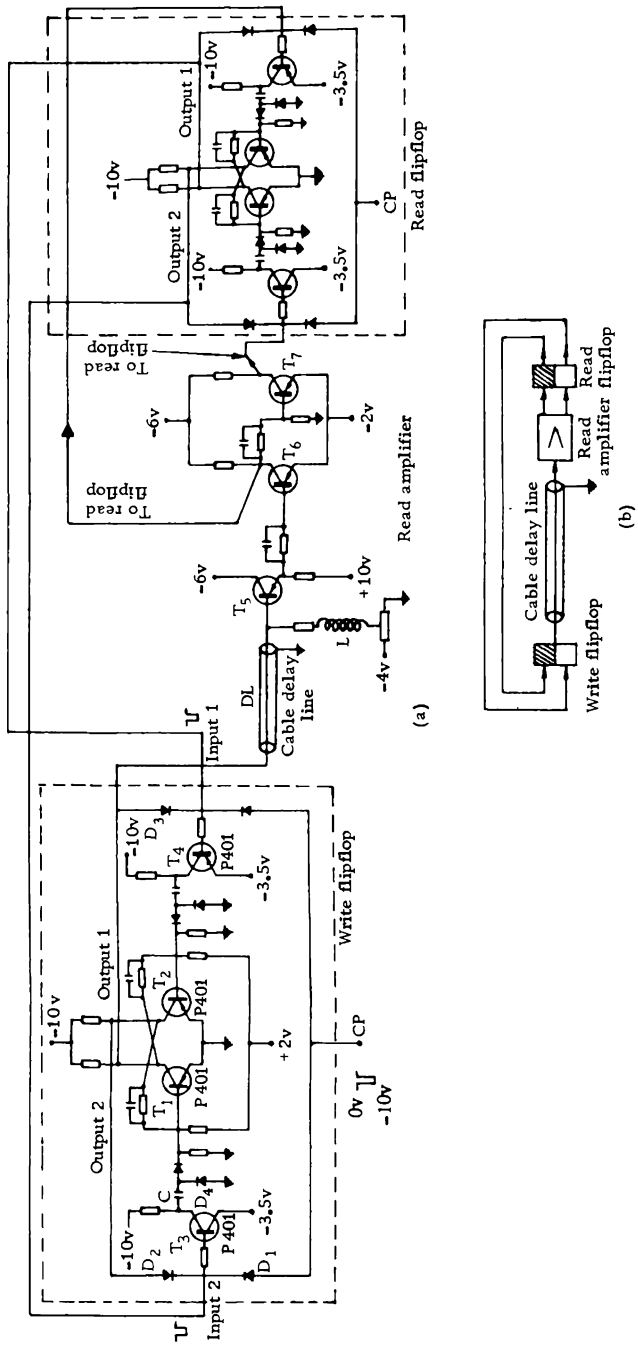


Fig. 250. Diagram of read-write setup in cable delay line: a—circuit diagram; b—block diagram.

the total delay time of the cable is 28/3 = 9.33 μsec. The delay in the read amplifier and in the two flipflops is 2/3 μsec; thus, the total delay time is 10 μsec, corresponding to the duration of a train of 30 pulses at a frequency of 3 Mc. The operating stability of the cable delay line is comparable with that of an ordinary electromagnetic delay line with lumped parameters. The read amplifer consists of two stages of amplification using transistors of types P-401 and P-403. The first stage (transistor T_5) is an emitter follower which isolates the cable line from the amplifier proper and gives a relatively high input resistance comparable with the characteristic impedance of the cable line (1350 ohms). The signal arriving at the base of emitter follower T_5 is of sufficient amplitude due to the low signal attenuation (6 db) in the line. The inductance L at the base bias supply increases the resistance shunting the base-emitter circuit of transistor T_5. In addition, the current in inductance L after passage of the pulse cannot fall rapidly to zero, which results in rapid switching of transistor T_5 and its fast cutoff after pulse passage. At the output of T_5 the pulse is first differentiated by R_1 and C_1 and then applied to the base of T_6. The second stage of the read amplifier contains two transistors T_6 and T_7 with inverted outputs to trigger the read flipflop. When a signal at one input of this flipflop cuts off one of its transistors, the signal at the second input simultaneously causes the other transistor to conduct, resulting in rapid switching of the flipflop to the new state. The read flipflop circuit is the same as that of the write flipflop. The coincidence circuit is shown in Fig. 251.

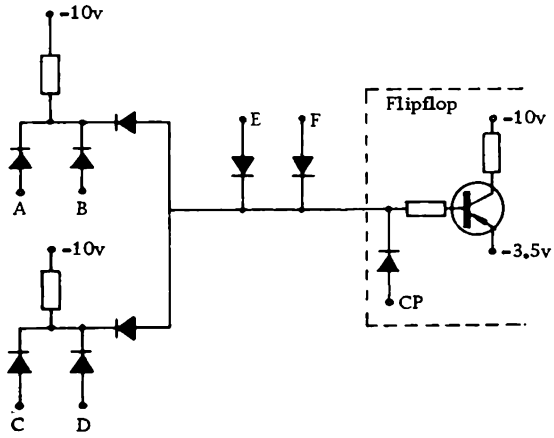


Fig. 251. Schematic of the coincidence circuit.

3. DIGITAL INTEGRATOR

The serial integrating unit discussed in Section 7 of Chapter I performs integration for all integrators by using one adder and by

extracting y and S number codes from the memory. The time required to obtain one problem solution (a single iteration) is $t \cdot n$, where t is the integration time of one integrator and n is the total number of integrators.

Problem solution speed can be considerably increased if integrations are performed in parallel by using many integrators. These integrators are interconnected and operate simultaneously in accordance with the problem solution in the same manner as in an analog differential analyzer. In a parallel integrating computer the problem solution time is decreased by a factor of n in comparison with a serial integrating computer. The problem solution time in a serial integrating computer increases in proportion to the number n of integrators required for the problem. The problem solution time in a parallel integrating computer is independent of the number of integrators and depends only on the time required for a single integration. This time, in turn, depends on the number of digits in the y and S registers and on the pulse frequency during serial addition (because the integration is still a serial operation). For example, in solving the equation

$$\frac{d^2y}{dx^2} + f(x)\frac{dy}{dx} + x \cdot y = 0 \qquad (3)$$

the integrators are interconnected in the manner shown in Fig. 23. For each increment dx, the $f(x)$ function generator unit delivers increments $df(x)$ to the integrators. Addition of all increments corresponding to the left members of equation (1) is achieved by a servo integrator $\sum$ with output feedback. Each of the parallel integrators has its own adders and other circuits for integration. The integrators are interconnected by direct connection of the output of one integrator (or several) to the dx input or dy input of other integrators. These interconnections are made on a plugboard during setup. The parallel integrating computer requires neither a common memory nor a common control unit, as are required in a serial integrating computer.

Each digital integrator of a parallel integrating computer performs the integration operation by a method close to the trapezoidal method and by ternary coding of increments ΔS, Δy and Δx. As shown in Fig. 252, it consists of two registers y and S, each of which stores a number code including the sign bit and the initial and final pulses. These registers may be coaxial cable segments or ordinary electromagnetic delay lines. The line delay is determined by the pulse repetition frequency and the number of bit positions. If the number of bits of the y or S register is 20 and the pulse repetition frequency is 2 Mc, each line must have a delay of 10 μsec. The higher the pulse repetition frequency, the shorter the delay line and, of course, the higher the required speed of the associated circuits. Other types of registers using transistors, ferrites, etc. may also be used.

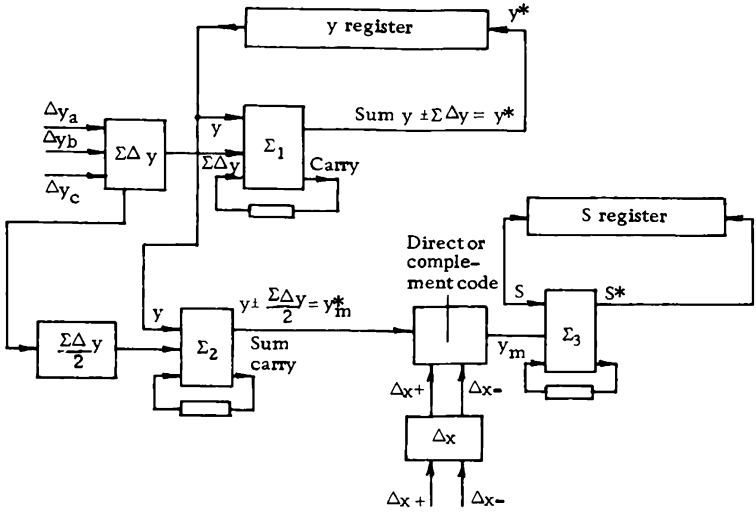


Fig. 252. A digital integrator using trapezoidal method.

For trapezoidal integration each integrator must have three adders and one $\Sigma\Delta y$ circuit to sum the increments from several input channels (see Chapter I, Section 3).

Figure 253 shows the detailed logic diagram of a single integrator. Simultaneous pulses from two input channels (inputs 1 and 2) can be received. Algebraic summing of the Δy increments in ternary coding is achieved by diode-transformer coincidence gates B_1 and B_6 and by inhibit gates B_2-B_5. If the increment pulses at both inputs correspond to +1 and +1 or -1 and -1, coincidence gate B_1 or B_6 causes output pulses to be stored as +2 or -2. If there is an increment at only one input (+1 or -1), gates B_2-B_5 cause output pulses to be stored as +1 or -1. However, these gates inhibit the input pulses if pulses at the two inputs are of different polarities (+1 and -1). The $\sum \Delta y$ increment pulses are stored with sign by dynamic flipflops FF_1-FF_3, which form a shift register. Coincidence gate B_7 inserts in the $\sum_1$ adder code 1 (for negative $\Sigma \Delta y$) in all remaining bit positions of the number code. The $\sum \Delta y$ number code is shifted to the adder by the CP pulses through gate B_8; these pulses occur in all bit positions beginning with the first pulse after the initial pulse H. The shift register is cleared by the last pulse P_n. The $\sum \Delta y$ code pulses are applied to the $\sum_1$ adder input after being delayed by one bit through delay line DL (beginning with the first pulse after the initial pulse) in order to obtain

doubling of $\Sigma\Delta y/2$ (i.e., $\sum \Delta y$ itself). Adder $\sum_1$ gives the new value of ordinate y^*,

$$y^* = y \pm \sum \Delta y, \tag{4}$$

which is serially transferred through gate B_9 to the y register (a delay line register). This gate passes all code pulses of y^* except initial pulse H, which is rewritten through gate B_{10}.

The number from the $\sum \Delta y$ shift register also reaches adder $\sum_2$, where it is summed with the y number code. Since the $\sum \Delta y$ code arrives at the adder without delay, it is summed with the first (least significant) bit arriving after the initial pulse. Thus, in distinction from adder $\sum_1$, adder $\sum_2$ gives the new value of mean ordinate y_m^*,

$$y_m^* = y \pm \frac{\sum \Delta y}{2}. \tag{5}$$

Depending on the increment $\Delta x+$ or $\Delta x-$ (+1 or -1), adder $\sum_3$ receives the direct or complement code of y_m, which is then summed with the number code stored in the S register to give a new value of S^*,

$$S^* = S \pm y_m^* \tag{6}$$

The direct or complement code of y_m is generated by gates B_{15}-B_{18} and dynamic flipflops FF_6 and FF_7. This scheme is the same as that in the serial integrating computer (see Chapter V, Section 3). The new S^* number code is transferred to the S register (a delay line register) through gate B_{19}. This gate passes all code pulses of S^* except the last pulse P_n, which is rewritten through gate B_{20}. The ΔS increment pulse at the output of adder $\sum_3$ is generated in ternary code by a circuit consisting of gates B_{21}-B_{22} and dynamic flipflops FF_8-FF_9, as was described previously (see Chapter I, Section 6).

The so-called servo integrator is often used in the solution of various problems. A servo integrator may consist of only adder $\sum_1$ and a y register. The S register and adders $\sum_2$ and $\sum_3$ are not needed in a servo integrator.

The servo integrator output can be obtained in a ternary form by using a circuit consisting of diode-transformer gates B_{11}-B_{14} and dynamic flipflops FF_4-FF_5. This circuit establishes whether or not there is an error code (stored in y register) and its sign. Whenever there is a code 1 in any bit of the y number code, it

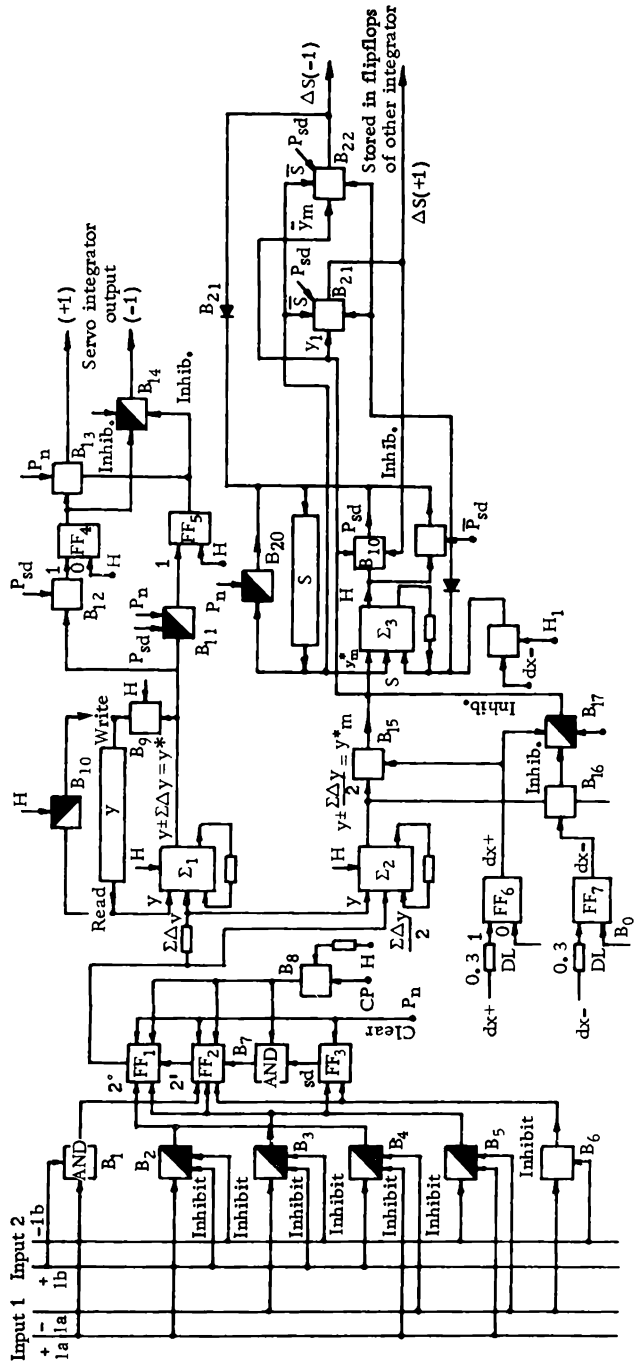


Fig. 253. Logic diagram of a digital integrator.

is stored in flipflop FF_5 through gate B_{11}. Sign code 1 of y (a positive number) is stored in flipflop FF_4; the absence of pulses at the FF_4 output corresponds to a negative number y. If code 1 is present in even one bit of number y, gate B_{13} gives an increment pulse corresponding to +1. The servo integrator output is connected with the suceeding integrators in the same manner as the increment output.

The above described digital integrator consists of approximately 25 gates, 8 dynamic flipflops, 3 adders and 2 10-μsec delay lines. This integrator requires approximately 25 transistors, 120 diodes, and 70 ferrite-core transformers. Integrator operation is controlled by the following timing pulses: initial pulse H, final pulse P_n, first pulse P_1 (the initial pulse delayed by 1 bit), and a pulse for the sign bit P_{sd}. The ΔS increment pulses and the servo integrator pulses are delivered to all integrators upon occurrence of last pulse P_n. All of the timing pulses are obtained from the CP pulse counter, which is common to all integrators of the parallel integrating computer.

4. MULTIPLIER INTEGRATOR

Multiplication of two numbers x and y in a serial integrating computer is achieved by means of two integrators and one adder. This method is not economical in a parallel machine (Fig. 253) since an increased number of integrators is required in a problem solution. It is more practical to use a special multiplier containing fewer elements than the multiplier using two integrators. Moreover, better multiplication accuracy can be achieved. The special multiplier is based on the following relation:

$$\Delta(x \cdot y) = x \cdot \Delta y + y \cdot \Delta x + \Delta y \cdot \Delta x = x \cdot \Delta y + y^* \cdot \Delta x, \tag{7}$$

where x is the old value and y^* is the new value of the multiplied numbers, and Δx and Δy are the increments. In comparison with the usual relation used in multiplication,

$$\Delta(x \cdot y) = x \cdot \Delta y + y \cdot \Delta x, \tag{8}$$

relation (7) is more accurate since the latter neglects the term $\Delta x \cdot \Delta y$ as shown in Fig. 254. The more accurate multiplication according to relation (7) requires the multiplication of the old value of x by increment Δy and multiplication of the new value y^* by increment Δx and addition of the two products in the S register, the overflow of which gives increment $\Delta(x \cdot y)$.

Since increments Δx and Δy are taken to be equal to unity, addition of $x \cdot \Delta y$ and $y^* \cdot \Delta x$ reduces to simple addition of x and y^* with consideration of the signs of numbers x and y^* and the signs of increments Δx and Δy. The rules for multiplication are given below.

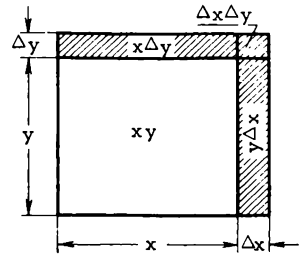


Fig. 254. Generation of product.

(a) The new values of x and y (equal to $x + \Delta x$ and $y + \Delta y = y^*$) are obtained by two adders in the same manner as in an ordinary integrator.

(b) Multiplication of the old value of x by the increment Δy and of the new value of y^* by the increment Δx with consideration of the signs are achieved by ordinary rules:

$$+x\cdot(-\Delta y) = -x\cdot\Delta y;\ (-x)\cdot(-\Delta y) = +x\cdot\Delta y;\ \text{etc.}$$

Positive numbers are expressed in direct code and negative numbers in complement code.

(c) Summation of $x\,\Delta y + y^*\cdot\Delta x$ is achieved by a third adder. There may occur overflow from the sum register and carry from the most significant bit. This overflow is the increment $\Delta(x\cdot y)$ of the sum of the products,

$$\Delta S_1 = \Delta(x\cdot y) = x\cdot\Delta y + y^*\cdot\Delta x.$$

The sign of the increment and the sign of the remainder in the sum register are determined according to the addition rule.

(d) A fourth adder is used for addition of the number code in the product register S and the number code from the previous addition (i.e., $x\cdot\Delta y + y^*\cdot\Delta x$). As a result of the addition $S = (x\cdot\Delta y + y^*\cdot\Delta x)$, there may occur overflow from the sum register and carry from the most significant bit. This overflow is the increment $\Delta S_2 = \Delta(x\cdot y)$. The sign of increment ΔS_2 and the sign of the number code in the S register are determined from the addition rules.

Let us give a few examples of the above rules.

Example 1

Old values	New values
$x = 6$	$\Delta x = +1;\quad x = 7$
$y = 6$	$\Delta y = +1;\quad y = 7$
$x\cdot y = 36$	$x^*\cdot y^* = 49$

In binary number system:

Code of x is [0] 0110	Code of new x is [0] 0111
Code of y is [0] 0110	Code of new y is [0] 0111
Product $x \cdot y$ [0] 0 0 1 0 [0] 0 1 0 0 (36)	

bits $2^7 2^6 2^5 2^4$ $2^3 2^2 2^1 2^0$

Product $x \cdot \Delta y$ is [0] 0110
Product $y^* \cdot \Delta x$ is [0] 0111
Sum $x \cdot \Delta y + y^* \cdot \Delta x$ is [0] 1101 Increment $\Delta S_1 = 0$; No overflow

Code in S register is [0] 0100
Code of sum $x \Delta y + y^* \Delta x$ is [0] 1101
Sum [0] 0001 Overflow $\Delta S_2 = +1$;
New product xy is [0] 0011 [0] 0001, i.e., 49.

Example 2

Old values		New values	
$x = -7$	$\Delta x = +1$	$x^* = -6$	
$y = +8$	$\Delta y = -1$	$y^* = +7$	
$xy = -56$		$x^* \cdot y^* = -42$	

In binary number system:

Code of x [1] 1001	Code of new x^* [1] 1010
Code of y [0] 1000	Code of new y^* [0] 0111
Product $x \cdot y$ [1] 1100 [0] 1000	Product $x^* \cdot y^*$ [1] 1101 [0] 0110

Product 111 carries
$x \cdot \Delta y$ is [0] 0111
Product +
$y^* \cdot \Delta x$ is [0] 0111
[0] 1110 $\Delta S_1 = 0$

Code in S register [0] 1000
[0] 1110
Code of $x \cdot \Delta y + y^* \cdot \Delta x$ [0] 0110 $\Delta S_2 = +1$

Example 3.

Old values		New values
$x = +8$	$\Delta x = +1$	$x^* = +9$
$y = -9$	$\Delta y = -1$	$y^* = -10$
$x \cdot y = -72$		$x \cdot y = -90$

In binary number system:

Code of x	[0]	1000			Code of new x^*	[0]	1001		
Code of y	[1]	0111			Code of new y^*	[1]	0110		
Product $x \cdot y$	[1]	1100	[1]	1000	Product $x^* y^*$ (see below)				

Product $x \cdot \Delta y$ is:	[1]	1000		
Product $y^* \cdot \Delta x$ is:	[1]	0110		
$x \cdot \Delta y + y^* \cdot \Delta x$ is	[1]	1110	$\Delta S_1 = -1$	

Code in S register

	[1]	1000		
Code $x \cdot \Delta y + y^* \cdot \Delta x$ +	[1]	1110		
	[1]	0110	$\Delta S_2 = 0$	

Code of product $x^* \cdot y^*$ is

	1 carry			
+	[1]	1100		
	[1]	1111		
	[1]	1011	[1]	0110
	or (−80)		(or −10)	Total −90

Fig. 255 shows the diagram of a multiplier integrator which is based on the above-described method. The integrator consists of three distributed-parameter lines x, y, and S, serving as the x, y and S registers, and four single-bit adders $\sum_1 - \sum_4$. Adders $\sum_1$ and $\sum_2$ perform addition of increments Δx and Δy to the x and y number codes arriving in sequence from delay lines x and y. The increments Δx and Δy (in ternary form as increments of +1 or -1) are stored by flipflops FF_1-FF_4. These increments arrive at the input of adder $\sum_1$ (and $\sum_2$) through gates $B_1 - B_4$ at the moment of passage from the x (and y) line of the first bit of the number x (and y) which follows after the initial pulse H is stored in flipflop FF_H. A pulse from the output of T_1 with a one-bit delay forms pulse H_1, corresponding in time to passage of the first (least significant) bit of the number x (and y). Gates B_5 (and B_6) inhibit rewrite of number x (and y) while flipflop FF_H is on; at that time the new number code from the output of adder $\sum_1$ (and $\sum_2$) is written in the x (and y) lines. The product $x \cdot \Delta y$ is formed by gates $B_7 - B_9$. Gate B_7 passes the direct code of number x to the input of adder $\sum_3$ if the increment Δy is positive (+1). If increment Δy is negative, B_8 and B_9 pass the complement code of number x. In this case, if a code in a given bit of number x

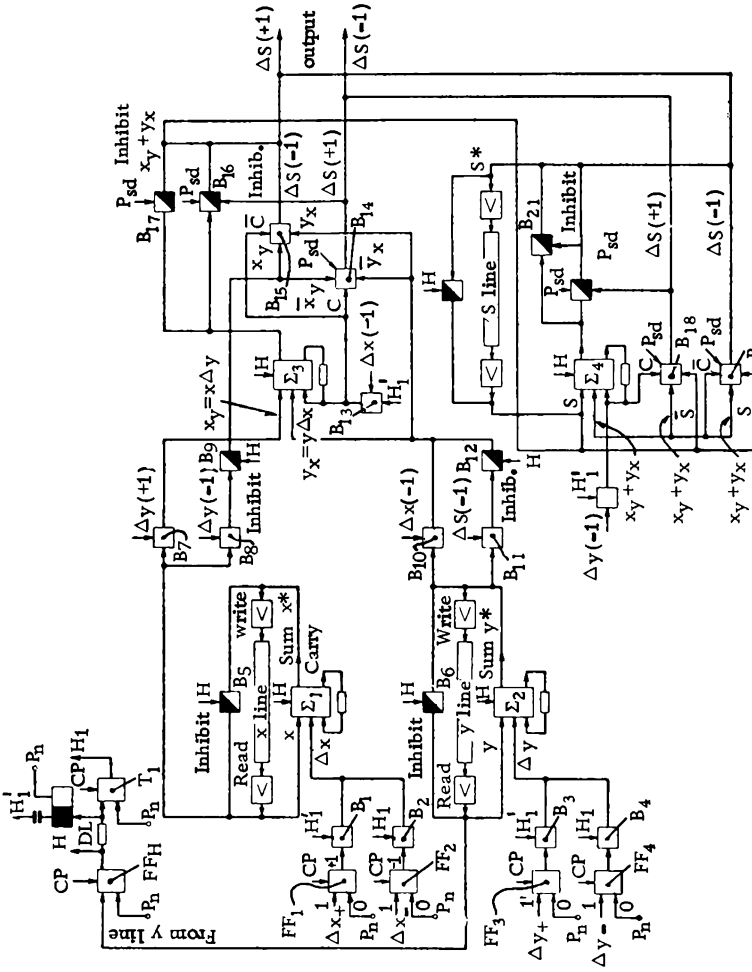


Fig. 255. Multiplier integrator.

is 0, pulse H passes through gate B_9 (from flipflop FF_H). If a code in the given bit of number x is 1, passage of pulse H is inhibited. The product $y^* \cdot \Delta x$ is formed by gates $B_9 - B_{11}$, which pass the direct or complement code of y^* to the adder input.

Addition of the resulting number codes x_y $(= x.\Delta y)$ and y_x $(= y^* \cdot \Delta x)$ is achieved by means of adder $\sum_3$. Since, for negative increments Δx and Δy, the 1's complement codes of the numbers are transferred to the input of adder $\sum_3$, it is necessary to add +1 in the first bit of the numbers being added to obtain the 2's complement code. This is achieved by gate B_{12}, which passes the first (after the initial) pulse H_1 in the presence of a negative increment Δx. In the event of a negative increment Δy, addition of +1 is achieved at another input of adder $\sum_3$ by means of B_{13}. From the output of adder $\sum_3$ the resulting sum $x_y + y_x = x \cdot \Delta y + y^* \Delta x$ is further transferred to adder $\sum_4$. When x_y and y_x are added, overflow of the sum register may occur and increments ΔS (+ 1) or ΔS (— 1) (the increment $\Delta (x \cdot y)$ of the product) is generated. If both x_y and y_x are positive and there occurs a carry C to the sign bit, increment ΔS (+1) is obtained by gate B_{14}. If both x_y and y_x are negative, increment ΔS (—1) is formed by gate B_{15} (see Chapt. I, Sect. 6). The correct sign of the sum $x_y + y_x$ is formed by gate B_{16}, which erases the sign obtained in the addition if a positive increment ΔS (+1) occurs. Gate B_{17} passes the entire sum code except the sign.

Addition of the number $x_y + y_x$ to the number S stored in line S is achieved by adder $\sum_4$. Here, an overflow may also occur, which corresponds to the product increment $\Delta S = \Delta (x \cdot y)$ and is +1 or -1. As in adder $\sum_3$, this increment is formed by gates B_{18} and B_{19}. Gate B_{20} erases the sign code of the sum, $S + (x_y + y_x)$, if increment ΔS is +1, while gate B_{21} passes the entire sum code except the sign. Increments ΔS (+1) or ΔS (—1) are delivered at the output of the multiplier integrator in parallel with the output lines ΔS from adder $\sum_3$ and are stored in flipflops at the input of the next integrator.

In entering initial data into the multiplier integrator it must be kept in mind that the numbers x and y must have the same scale and that the initial pulse for both numbers must be written in the same bit position. Increments Δx and Δy are applied to the input of multiplier integrator in the form of units code pulses.

5. PULSE INCREMENT ADDER

The addition of increments at the integrator input (with more than two input channels) is achieved by a special circuit. Fig.

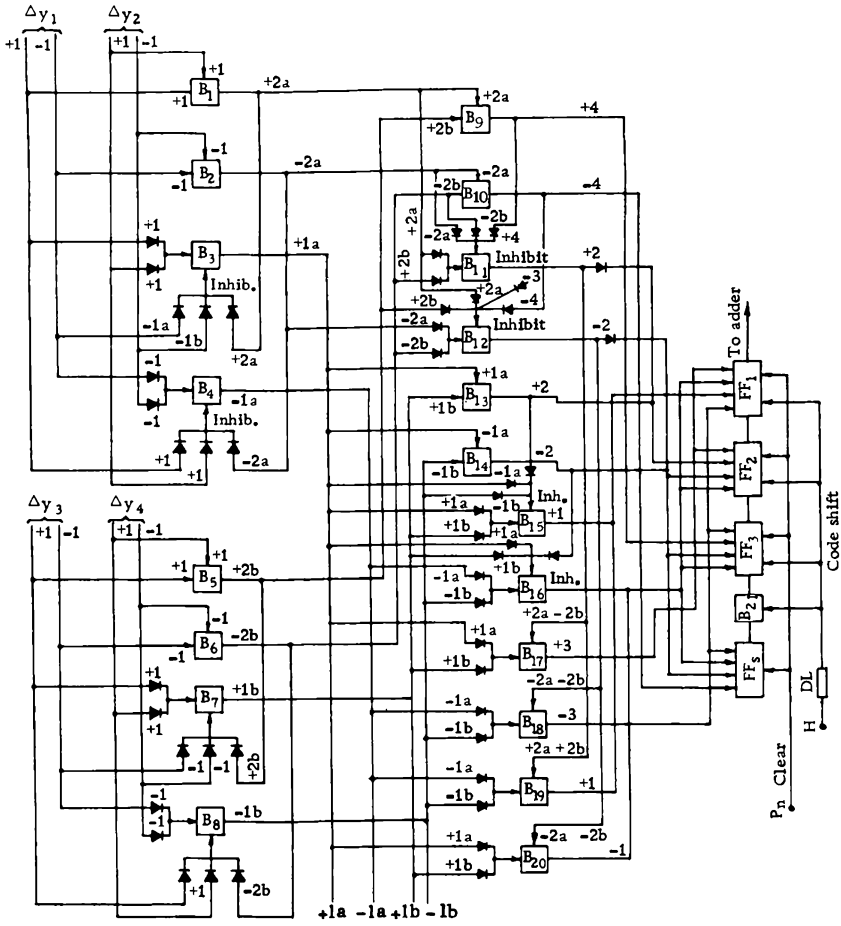


Fig. 256. Diagram of increment addition.

256 shows a two-stage increment adder for four input channels at which increments Δy_1, Δy_2, Δy_3 and Δy_4 in ternary coding arrive. Increments Δy_1 and Δy_2 are summed in one circuit of the first stage and increments Δy_3 and Δy_4 in the other circuit of the first stage. Increments Δy_1 and Δy_2 are added by four gates $B_1 - B_4$. Gates $B_1 - B_2$ deliver an output pulse only if both increments Δy_1 and Δy_2 are +1 or -1 (i.e., their sum is +2 or -2). Gates B_3 and B_4 deliver an output pulse if the sum of increments Δy_1 and Δy_2 is +1 or -1 (i.e., a unit increment at one of the inputs). Passage of +1 or -1 through gates B_3 and B_4 is inhibited if the sum of the increments is equal to two units or if the increments have different polarity, resulting in a zero sum. In short, gates $B_1 - B_4$ produce an increment sum equal to one of the following values:

+2, -2, +1, -1. Operation of the other circuit for increments Δy_3 and Δy_4 is similar.

The second stage of increment addition uses gates B_9–B_{20}, by which various combinations of pulses are achieved at lines +2a, -2a, +1a, -1a and +2b, -2b, +1b, -1b. If the increments at all four inputs are +1 or -1, number code +4 or -4 is obtained by means of gates B_9 or B_{10} as a result of coincidence of pulses at lines +2a, +2b or -2a, -2b. Gates B_{11} and B_{12} form number code +2 or -2 if there is a pulse at only one of lines +2a or +2b (-2a or -2b). These gates are cut off by inhibit pulses if code +4 is formed or if increments of opposite polarities occur (e.g., increments at lines +2a and -2b or -2a and +2b). When unit increment occurs at any two inputs of the adder, +2 or -2 is generated by gates B_{13} and B_{14}. Gates B_{15} and B_{16} generate number code +1 or -1 if there is a unit increment at only one of the four inputs. If increments of opposite polarities occur or if two increments of the same polarity (+2 or -2) occur, passage of increments through gates B_{15} and B_{16} is inhibited. Gates B_{17} and B_{18} form the increment sum code of +3 and -3. Code +3 corresponds to the presence of an increment at lines 2a or 2b and simultaneously of an increment of the same sign at lines 1a or 1b. Finally, gates B_{19} and B_{20} form the sum code of +1 or -1 when there is an increment at lines -2a, -2b or +2a, +2b and there is simultaneously obtained an increment of opposite sign at lines +1a, +1b or -1a, -1b.

The increment sum code formed by one of gates $B_{19} - B_{20}$ is stored in a shift register consisting of three dynamic flipflops FF_1-FF_3 and a sign flipflop FF_S. Positive numbers are stored in the register in direct code and negative numbers in complement code. The number stored in the register is shifted by pulses H after one-bit delay by line DL from a flipflop triggered by the initial pulse of each number. This flipflop delivers pulses coinciding in phase with the CP pulses up to the last pulse P_n, by which the flipflop is set to the 0 state. Gate B_{21} transfers sign code 1 to the adder in all bit positions in order to obtain the complement code for a negative number. The shift register is cleared by the last pulse P_n.

6. DIGITAL SERVO

The digital servo in a parallel integrating computer is a separate element similar to the digital integrator in a serial computer. It is used as a comparator or a null device in problem solution or as an element in the generation of nonlinear or discontinuous functions. If the servo is used to generate the difference (or error) $\Delta = A - B$ of two compared numbers A and B, increments ΔA and ΔB are the inputs, and the output is fed back to the input until the difference Δ in the y register of the integrator becomes zero.

In distinction from the servo integrator in a serial integrating computer (see Chapt. I, Sect. 10), the digital servo in a parallel computer (Fig. 257) for economy of elements consists of only one register y. There are only two inputs Δy_1 and Δy_2 in ternary coding.

If more inputs are required, a special code-pulse adder is used. The ΔS input of the servo is also generated in ternary coding in the form of a positive, negative or zero increment.

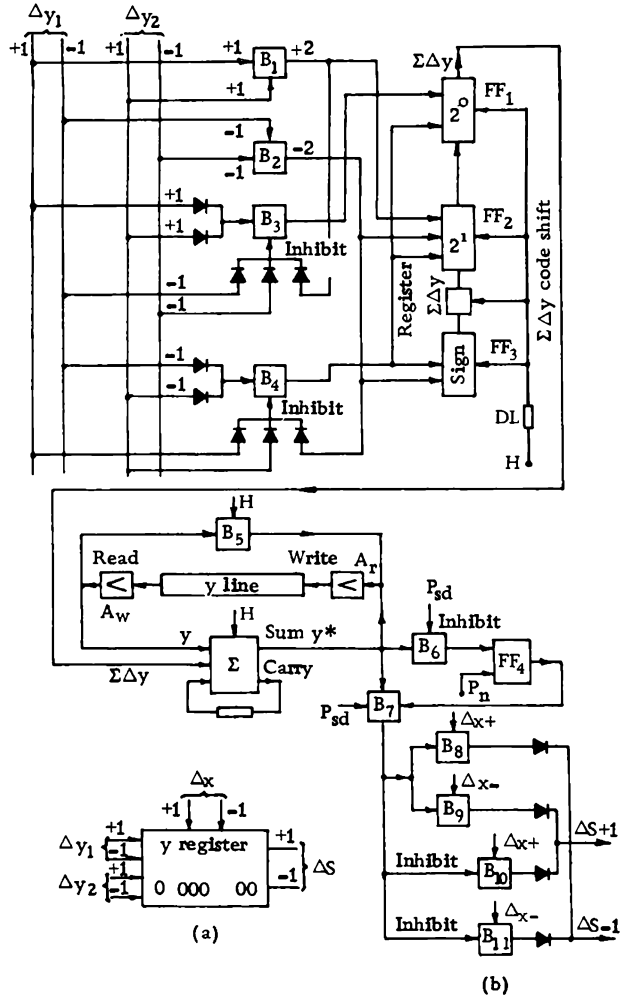


Fig. 257. Digital Servo with two inputs.

The digital servo contains the following circuits:

(a) Gates B_1 — B_4 for addition of the code pulses at two inputs Δy_1 and Δy_2;

(b) A register (flipflops FF_1-FF_3) storing the $\sum \Delta y$ number code;

(c) A y register in the form of a cable delay line with read and write amplifiers A_r and A_w;

(d) A single-bit adder $\sum$ for addition of numbers y and $\sum \Delta y$;

(e) Gates $B_6 - B_{11}$ and flipflop FF_4 to deliver ΔS code pulses at the output.

Gates $B_1 - B_4$ receive the Δy_1 and Δy_2 code pulses arriving at the two pairs of input lines and generate a binary code corresponding to their algebraic sum. If both inputs Δy_1 and Δy_2 are positive increments (+1), coincidence gate B_1 passes a pulse to flipflop FF_3 corresponding to the second bit of the $\sum \Delta y$ register. If both inputs are negative increments (-1), gate B_2 delivers to the $\sum \Delta y$ register the complement code corresponding to number -2. The two other gates, B_3 and B_4, transmit increment +1 or -1 occurring at either input Δy_1 or Δy_2. If pulse coincidence occurs at the outputs of gates B_1 and B_2, inhibit pulses are applied to gates B_3 or B_4. The number code $\sum \Delta y$ is transferred by shift pulse H to the serial adder. The y register is a section of coaxial delay cable (type RKZ-401, see Sect. 2). The number code stored in the y register is transferred to the other input of the adder. The pulses arriving from the delay line are shaped by amplifier A_w and amplified by amplifier A_r. The summation $y \pm \sum \Delta y \; (= y^*)$ is performed in adder $\sum$. The new sum y^* is entered into the delay line to replace the old code y through gate B_5, which is unblocked by pulse H at the beginning of the add cycle.

The ΔS increment at the servo output is generated at two lines in the form of code pulses +1 or -1 or 0 (the absence of pulses) in the following manner:

(a) If the number code of y^* (except the sign) is other than zero, there is an increment at the output, as this signifies the existence of a difference stored in the y register. In addition, there must be an increment of the independent variable x. In a special case, the increments of the independent variable are timing pulses. The absolute values of the increments are:

$$\left.\begin{aligned} |\Delta S| &= 1, \\ |\Delta x| &= 1, \\ |y| &> 0. \end{aligned}\right\} \tag{9}$$

(b) The sign of increment ΔS (+1 or -1) is determined by the signs of both number y and the increment Δx (i.e., in a manner similar to the sign of the product of y and Δx). Thus,

$$\operatorname{sign} \Delta S = (\operatorname{sign} \Delta x) \cdot (\operatorname{sign} y). \tag{10}$$

The logic circuit implementing these rules consists of gates $B_6 - B_{11}$. Number code y^* at the adder output is applied through gate B_6 to the input of flipflop FF_4. If even one of the bits of number y (except the sign) is 1 (i.e., $|y| > 0$), flipflop FF_4 is set to the state corresponding to code 1. Gate B_6 is cut off during

passage of the sign bit of number y^*, while gate B_7 transmits the sign code of y^* if flipflop FF_4 is on (i.e., y is not zero).

Depending on the increment Δx (+1 or −1), gates B_9 and B_{10} generate a positive increment ΔS while gates B_8 and B_{11} generate a negative increment. For example, if $\Delta x = -1$ and the sign code of y^* is 1 (y negative), the increment ΔS is positive and gate B_9 transmits a pulse to the ΔS +1 output line. If y^* is a positive number and the increment Δx is negative, gate B_{11} transmits a pulse to the ΔS -1 output line.

7. INSERTION OF INITIAL DATA

The insertion of initial data in each integrator requires a register for storage of the initial value of the integrand y. For each multiplier integrator, two such registers are required. With a large number of integrators in a computer, the amount of electronic equipment required for these registers is considerable. It is more practical to use a common memory for storage of the initial values of all integrators. This memory must provide for readout as well as for writing of new initial values obtained externally, for example, as the result of computations in another digital computer or the data from external data transducers or the initial data from a keyboard. The initial data unit (Fig. 258) consists of one ferrite-core memory matrix having 100 rows (or words) with 26 bits in each row (depending on the number length used in the computer). Each row contains 26 ferrite cores for storing one number. Each core is threaded with one horizontal line for selecting the word line, one vertical line for read operation, and another vertical line for write operation. The vertical line for write operation inhibits writing in the core if code 0 is to be written, as follows. Each bit position of the write register in which the number being written is stored controls an anticoincidence gate which passes the write signal if the given bit of the register contains code 0 or does not pass it if this bit contains code 1. The write signal after amplification is applied to the vertical line of the matrix. The flux in the core switched by the current in the vertical line is subtracted from the flux switched by the write current in the horizontal line; this results in the core not being remagnetized.

For read operation the second vertical line is connected to a read amplifier which, after sensing the stored information in the core upon the read signal, transmits the code to the read register.

The address lines (or word lines) of the matrix are selected by means of a diode-transformer matrix. In order to reduce the number of components, the selection circuits for the 100 address lines of the memory matrix are divided into four groups with 25 address lines each. The diode-transformer matrix also has 25 output lines, each of which is connected in parallel to the corresponding address line in each group of the memory matrix. Selection of a group of address lines is achieved by gates $B_1 - B_4$, which in turn are controlled by the group number counter. The diode-transformer matrix is controlled from two series-connected

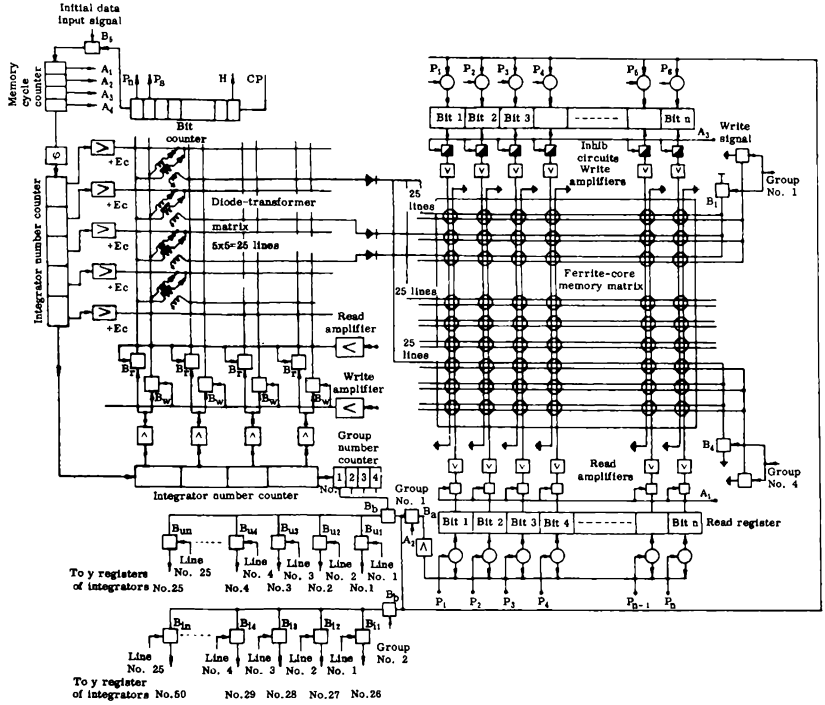


Fig. 258. Ferrite-core memory for initial data storage.

integrator-number counters, each having five stages. One counter drives the horizontal lines and the other the vertical lines of the diode-transformer matrix through power amplifiers. The read and write operations in each address line require different current polarities. For this purpose, each transformer of the diode-transformer matrix has two primary windings with opposed turns and one secondary winding in which the current direction depends on which of the primary windings is pulsed. Each of the primary windings is connected to a separate vertical line. There are twice as many vertical lines as horizontal lines in the diode-transformer matrix. The two vertical lines are driven by a single amplifier controlled by one counter stage. Switching of the vertical lines from write to read and vice versa is achieved through read gates B_r and write gates B_w. For control of number entry the clock pulses CP arrive first at a bit counter consisting of 26 stages. This counter delivers timing pulses corresponding to each bit of the number, including initial (start) pulse H, sign bit pulse P_S and last pulse P_n. These pulses are also used for control of operations in the integrators. Through gate B_5 (controlled by the initial-value input signal), a pulse from the bit counter is further transmitted to a memory cycle counter consisting of four stages $A_1 - A_4$. Stage A_1 delivers a timing pulse for the start of readout of the number code from the memory matrix to the read register

and, for connection of the read amplifier, stage A_2 delivers a timing pulse to shift the number code from the read register to the appropriate integrator through gate B_a. Simultaneously, the read-out code is transferred to the write register for rewrite into the memory matrix.

Stage A_3 delivers a timing pulse for connection of the address write amplifier and for application of the write signal to the anti-coincidence gates. Through the amplifiers these gates deliver the complement code in the write register to the vertical inhibit line of the memory matrix. Stage A_4 provides a timing interval sufficient for damping of transients during the write operation before beginning another read operation. Thus, the read-write operation of one word takes four timing intervals of the bit counter. If the CP pulses have a repetition frequency of 3 Mc, the timing interval for serial readout of a 30-bit number is 10 μsec and the read and write cycle time can be 40 μsec. The pulses from the memory cycle counter are further transmitted to the integrator number counter and then to the group number counter. The group number counter consists of four stages which control gates B_1 — B_4 of the memory matrix. The number code from the read register is transferred through gate B_a to the y register (the delay line of the corresponding integrator) at the CP frequency. The number code is transferred to the appropriate integrator by gates B_{u1} — B_{un}, which are controlled by the corresponding integrator number lines. Gate B_b, passing the number code to the appropriate integrator group, is controlled by one of the four stages of the group number counter.

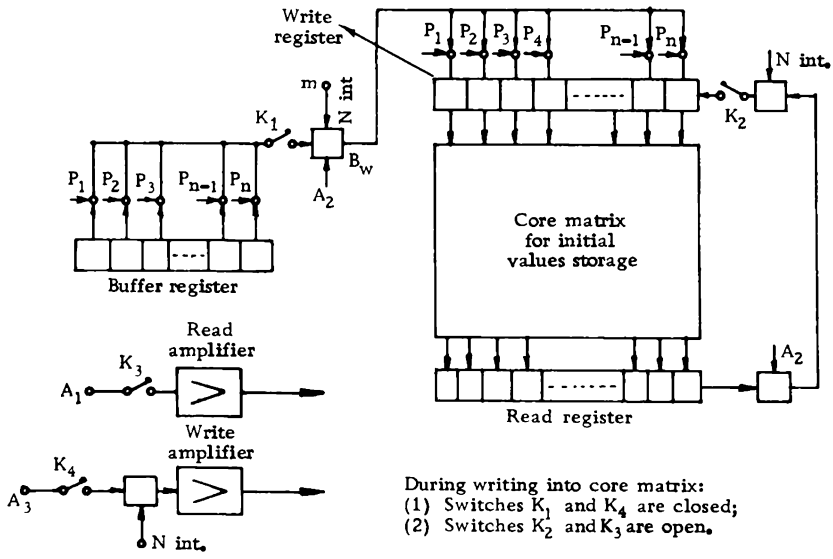


Fig. 259. Entry of external data into the memory matrix.

If a number arriving from external channels need only be written in the core matrix, the number is first stored in a buffer register (Fig. 259) and then transferred serially to the write register through gate B_w, which is controlled by the integrator number line and the timing pulse from stage A_2. The number in the write register is written into the core matrix upon the timing pulse from stage A_3, which connects the write amplifier. A number can also be written into the core matrix from a keyboard, which enters the chosen code into the buffer register. The integrator number line (the word line) is then connected to gate B_w (to point m) and the number code is serially transferred to the write register. During the write operation switches K_1 and K_4 are closed and switches K_2 and K_3 are open.

Chapter X

Digital Integrating Computer Applications

1. AUTOMATIC CONTROL OF PRODUCTION PROCESSES

Integrated automation of such industrial activities as machine-tool operation, process control, production inventory and the like, requires the use of a digital control computer. Despite the variety in production processes and technology, the functions performed by the control computer are completely identical. Such computers must meet the following requirements:

(a) Possess high reliability;

(b) Perform computations for a large number of process parameters both interdependent and independent;

(c) Solve the problem of process optimization (maximizing a function of one and several variables);

(d) Possess a high computing speed (required for solution of many real-time problems including integration, differentiation, computation of polynomials, correlation analysis, etc.);

(e) Be simple in design and economical in operation;

(f) Solve the problem of dynamics of control processes described by systems of linear or nonlinear differential equations and integral equations (equations of the Fredholm type);

(g) Possess large memory capacity (this may be of the order of 1,000,000 bits for various logic and control functions including correlation analysis, smoothing and storage of the variables characterizing a process, storage of calibration data for measuring instruments and transducers, storage of data concerning the limits of changes in variables);

(h) Provide for multichannel data entry of functions in the form of graphs and tables obtained both empirically and by computation;

(i) Perform various logic operations during the control process that can be reduced to the elementary logic operations of AND, OR, NOT and their combinations;

(j) Perform the function of digital servos for continuous control of driving mechanisms with consideration of requirements on control dynamics and dynamic accuracy;

(k) Compare computed quantities with quantities measured in the process of continuous control;

(l) Achieve automatic alteration of the control program or computing program in accordance with given parameters;

(m) Provide for multichannel connection with transducers, measuring devices and actuating mechanisms of the controlled process. It must achieve multichannel conversion of analog quantities to digital and vice versa, linearization of input data, calibration of these data. For example, if A is proportional to the square root of the measured pressure P (i.e., $A = k\sqrt{P}$), the quantity A must be entered directly into the computer in accordance with the measured value of P. In addition, the computer memory must store data concerning the limits of change in the variables of the production process or data concerning their expected values. These data must be periodically compared with the current values and the error signal between them used for control of the production process. The error signal may be time integrated by a digital integrator to give an additional control function which can decrease the steady-state error.

The specific influences of many parameters of the production process on the control system, its operating characteristics, and output of the final product require the use of correlation analysis and the determination of maximum values for process characteristics or process optimization. The process parameters must be compared with the determined maximum values or with the given optimum criteria.

In order to determine the optimum operating conditions in a control process, digital simulation of the process may be required independently of the normal function of process control but with consideration and selection of data obtained as a result of modeling. Hence the control computer must provide special channels for parallel modeling of the current processes with a certain lead time relative to the latter in order to use the results of such modeling in process control. Of course, the functions of automatic monitoring of various processes must also be performed by the computer. The above listed functions for control of machine tools, engines, and technological processes throughout a shop (or even an entire plant) can be conveniently achieved from one central control location by means of a control computer. This is dictated primarily by economic considerations. Individual equipment for control of the machine tool or assembly will cost much more than a single control computer. The operation of large amounts of electronic equipment dispersed at various locations in an automated shop also requires considerable expenditures and numerous skilled personnel for supervision and maintenance. For example, programmed control of a single milling machine requires relatively complex electronic equipment with 100-150 vacuum tubes. The presence of a few score such machines requires the use of several thousand vacuum tubes, whereas one control computer may have but one tenth as many tubes and other electronic equipment. However, centralized control imposes more rigid requirements on reliability of the control computer, which may be achieved by using

semiconductor elements and, where necessary, duplication of equipment.

The concepts of centralized automatic control have already been applied in the control of moving objects (rockets, aircraft, ships), wherein numerous computing systems for various control functions are replaced by a single digital computer.

In the "Mars 1000" computer developed in the USSR for monitoring of many parameters of a technological process, the use of centralized control has permitted replacement of several hundred expensive control and measurement devices by a single computer which not only automatically measures numerous parameters but also prints out the results of computations and signals the occurrence of irregularities of the parameter under control. Such a control computer must not only monitor but also automatically control numerous objects and parameters of the production process. Two types of computers may be used for various computing and control functions: the digital arithmetic computer (DAC) and the digital differential analyzer (DDA). Depending on the application, both types can be general-purpose or special-purpose computers. The general-purpose digital differential analyzer is a large-memory computer which can solve the most varied mathematical problems by methods of integration.

It is evident that for the purpose of automatic control it is necessary to have special digital computers developed to meet the above listed requirements. The criteria for comparison of the DAC and DDA in control applications are the computing speed (see Table 14), reliability, cost and operating convenience, The problem solution rate in a control system must be high enough to handle the highest input data rate. In any case, the computing rate must be several times (from 4 to 8) higher than the maximum input data rate. Moreover, the initial solution of the problem (initial values) must be obtained without excessive time lag, since the input parameters may change considerably during a single computing cycle. Comparative data on the time required for execution of basic operations in the DAC and DDA with a magnetic drum memory of the serial type are listed in Table 14 [Ref. 13].

It is seen from Table 14 that the problem solution time in the DDA is at least one order of magnitude shorter than that in the DAC under identical conditions. Where the DAC delivers 10-20 solutions per second, the DDA delivers 100-200. This means that a DDA of this type with a magnetic drum memory is suitable for process control requiring a computing frequency of 0 to 50 cps, while the arithmetic machine is suitable only for control of processes in which the computing frequency does not exceed 5 cps. However, the DAC provides less delay at the beginning of problem solution, and a solution may be obtained after one computation cycle. The DDA requires a large number of cycles (iterations), depending on the required accuracy and on the inserted initial conditions. A decrease in delay time in the DDA requires an increase in iteration rate, primarily by increasing the CP rate.

TABLE 14

Execution Time of Basic Operations in DDA and DAC

Operation	Execution Time of Operations (time for one addition operation is taken as unity)		
	DDA	DAC	Remarks
Addition or subtraction	1	$1+T$	T, access time
Multiplication	2	$n+T$	n, no. of bits (15-20)
Division	4	$n+T$	
Sine or cosine	2		Up to 20 instructions with several multiplications
Integration	1		Several scores of operations

Thus, the DDA has limitations on the change in the process variable, while the DAC has a limitation on the number of variables. In other words, although the DDA may operate with many small increments per unit time, their total increment is relatively small. The DAC, on the other hand, may operate with large increments, but the execution rate is limited. In computation of the variables associated with a continuous process, the DDA requires considerably less time for algebraic and trigonometric operations than the DAC. Since, in the control of production processes, the input quantities vary continuously, the convenience of using a DDA is evident. Operation of the DAC is not based on continuous change in the variables of the controlled process since, as a rule, each cycle of computation does not employ the data obtained from the previous cycle. However, at the beginning of computation the DAC may deliver the result of solution more rapidly than the DDA. With regard to reliability and convenience of operation, the DDA contains less equipment than the DAC and, consequently, is less subject to breakdown. The number of incorrect solutions as a result of random errors is comparable in both types of computer. Although random errors have a tendency to accumulate in the DDA, they are generally small due to the extremely small value of the increments. Thus, despite a few disadvantages (e.g., the delay time at the beginning of problem solution), the DDA is more suitable for process control purposes, due chiefly to the high rate of problem solution and the reliability of the computer itself.

Digital integrating computers fully satisfy the above listed requirements for a control computer. In combination with digital arithmetic computers, they are able to solve the most complex process control problems. The high solution speed of a DDA permits serial solution of control problems for a large number of machine tools and assemblies by using one electronic apparatus and an electronic switching network. For example, if control of a

universal milling machine requires 30 integrators with an integration time of 20 μsec, then each cycle of iteration for one machine requires 600 μsec. In sequential control of 100 machines, the total time for one iteration is 0.06 sec. Thus, the frequency of control pulses is 16 cps, which is adequate for continous control of the machines.

During process control, the y register of the internal memory of a digital integrating computer may be used for the following purposes:

(a) Multichannel conversion of voltages to a digital form as increments with an accuracy of 0.01%,

(b) Comparison of the computed and measured quantities;

(c) As a reversible counter for pulses obtained simultaneously from several input channels;

(d) Summation of pulses with consideration of their sign for several channels;

(e) As a digital servomechanism (servo integrator);

(f) As integrators for the computation and interpolation of the given functions;

(g) As a generator of functions of one or several independent variables given in the form of experimental graphs or tables. In particular, the function generators may be used to generate profiling instructions for a milling machine;

(h) Execution of control logic in accordance with the given program.

Elementary operations may be provided for in the computer to alter the control function during the process by automatic switching. In particular, the computer may execute a conditional or an unconditional transfer.

Analog computers cannot fulfill the requirements of a process-control computer for the following reasons:

(a) They do not provide for storage of the required amount of data since they do not have a memory.

(b) The computational flexibilities of analog computers are extremely limited. Their use for solution of nonlinear problems is limited since even a relatively small number of nonlinearities leads to a sharp increase in the number of nonlinear elements (multipliers, function generators, etc). In addition, these nonlinear elements cannot provide the necessary accuracy of computation and require special elaboration in circuitry.

(c) Analog computers cannot perform programmed logic operations. Initialization to a new program of computation is inconvenient. In addition, they contain no facilities for automatic checking of computations or for error detection by the operator.

(d) From the standpoint of economy, analog computers are less desirable than digital computers due to the complexity in manufacture and the necessity for adjustment and calibration of individual elements, except those cases where the computation is quite simple and requires an accuracy of several tenths of a percent to one percent.

For operation in complex control systems the analog computer is less reliable than, for example, the digital differential analyzer. The sole advantage of analog computers is the handling of input data, most of which are measured in the form of continuous variables (voltages, angular displacements, etc). However, the technique for multichannel conversion from analog to digital form and vice versa makes this advantage less important.

2. DIGITAL PROGRAMMED CONTROL OF MACHINE TOOLS

The possibility of using the DDA as a control computer finds a successful application in the programmed control of metal-cutting machines, particularly the universal milling machine for machining complex curvilinear surfaces with high accuracy (of the order of a few microns). There are two methods of controlling machine tools by means of digital computers and DDA. In the first method, data obtained from a sketch of the article to be produced are used in compiling a computer program for insertion into the digital computer; this program is recorded on punched tape. For example, for production of the article sketched in Fig. 260 the data characterizing the profile of the article (straight lines, points of transition of straight lines to curves, radii of curvature, tolerances, etc) are converted to digital form as shown in Table 15.

TABLE 15

Initial Data

Sectors	Boundary of coordinate sector			Tolerance (mm)	Feed rate (cm/min)	Radius of curvature (mm)	Cutter radius (mm)
	x	y	z				
Start	1,500	1,000	5,000	0.025	---	---	125
A, B	3,500	1,000	5,000	0.025	125	---	125
B, C	4,556	5,287	5,000	0.025	125	2,250	125
C, D	1,000	7,128	5,000	0.025	125	---	125
D, E	1,000	1,500	5,000	0.025	125	---	125
E, A	1,500	1,000	5,000	0.025	125	0.500	125

From this table a process chart is prepared. Straight line passes are described on the chart by their end coordinates. Circular arcs are approximated by a number of straight line segments with indication of the end coordinates of the arcs, their radii and the required precision. For cutting concavities, it is necessary to have data concerning the outlines of the depression, the depth of the final cut and the number of rough cuts. The center

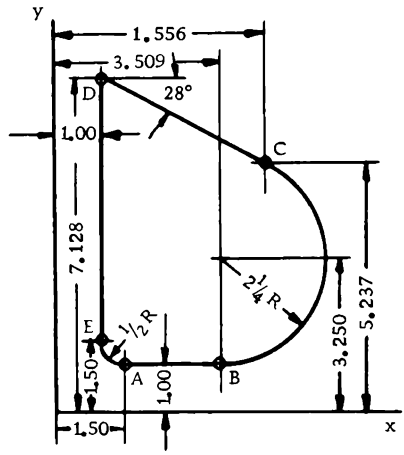


Fig. 260. Example of an article profile.

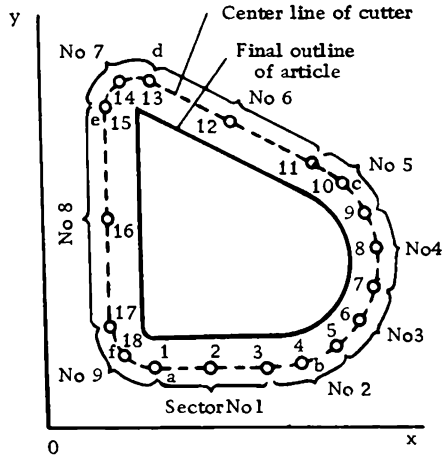


Fig. 261. Center line of cutter movement.

line of the cutter movement (Fig. 261) is divided into a number of sectors for which reference points are given. After filling out the process chart, a punched tape is prepared for the digital computer. The punched tape contains the coded representation of all data required for computation of the cutter path. The computer calculates the path of the tool in the form of increments of coordinates $x(t)$, $y(t)$, and $z(t)$ as functions of time. The cutter path is approximated by linear or parabolic sectors with the required accuracy. The feed rate of the tool is automatically determined for each of the coordinates by a count of the increment pulses over

a definite time interval. It is decreased in approaching points where there is a change in cutting direction and is so chosen that the cutting tool operates without overload. Since cutting according to the given profile requires displacement of the cutter center along the curve *abcdef* in Figure 261 (known as an equidistant curve) differing from the given contour by an admount d, the computer must compute the coordinates $\bar{x}$, $\bar{y}$, $\bar{z}$ of the equidistant curve of each pass rather than the coordinates of the article. These coordinates may be computed, for example, from the formulas (for a plane contour):

$$\bar{x} = x \pm \frac{y'd}{1+(y')^2}; \tag{1}$$

$$\bar{y} = y \mp \frac{d}{1+(y')^2}. \tag{2}$$

With a correction for tool wear, these formulas become

$$x^* = \bar{x} \pm \frac{y'\Delta}{1+(y')^2}; \tag{3}$$

$$y^* = \bar{y} \mp \frac{\Delta}{1+(y')^2}, \tag{4}$$

where x and y are the given coordinates of the article; y' is the derivative with respect to x; Δ is the change in cutter radius from the initial value. From the given table of coordinates x, y, z (or their increments), the computer computes the increments of the coordinates as functions of time by means of approximate formulas (polynomials) for the smaller linear sectors of the cutter path and the computed results are placed on punched tape. Since any trajectory of a cutting tool may be described by differential equations of its motion as a function of time, the computer may be a DDA which delivers unit increments $x(t)$, $y(t)$, $z(t)$ and approximates the segments of a curve in linear or parabolic form.

For example, as a digital arithmetic computer for machine-tool control we may use the serial computer G15D (Bendix Corporation) with a magnetic drum storage. The punched tape, the output medium of the computer, usually has 5-7 tracks. Three of these tracks contain the unit increments of coordinates $x(t)$, $y(t)$, $z(t)$ for each step (e.g., 5-10 microns). One track is used to record the feed rate of the tool and one track for coding the end of a pass. The remaining tracks are used as markers and for parity checking. The data from such a control tape is read by a photoelectric reader and transferred to the machine control unit. This control unit contains an internal memory in the form of registers, counters, etc., circuits for comparison of the computed and measured values of displacements and rates of the cutting tool in three coordinates, digital-to-analog converters, and servo controls for the horizontal, vertical and lateral displacements of the cutters.

As an example, Fig. 262 shows the block diagram for control of one coordinate of cutter displacement. Pulses read from the

punched tape (an aluminum foil reinforced by plastic coating on both sides) are transmitted from the read amplifier to displacement and rate counters. The increment pulses for coordinate x are accumulated in the displacement counter, forming the number code of coordinate $x = \sum \Delta x$.

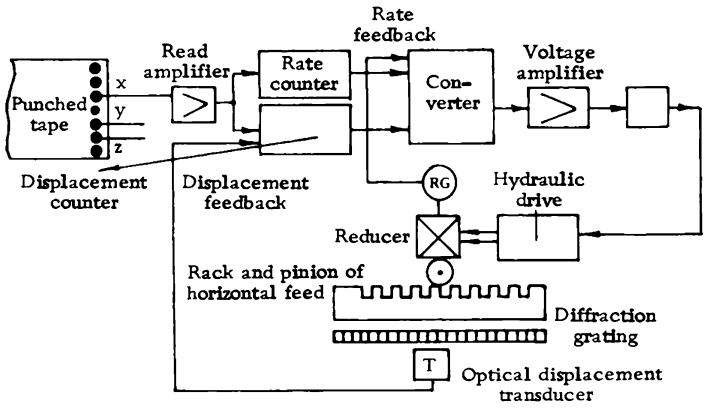


Fig. 262. Control of one coordinate of cutter displacement.

The displacements, measured at the machine tool by means of a displacement transducer, are fed back to the input of this same counter in pulse form. Thus, the contents of the counter show the difference Δ (or the error) between the assigned and measured values of displacement. The Δx increment pulses are also summed in the rate counter for a definite time, forming a number code proportional to the assigned rate of displacement. This code and the Δ displacement error code are applied to a digital-to-analog converter. The resulting voltage, proportional to the assigned rate of displacement, is compared with a feedback voltage which is proportional to the measured rate of displacement of the tool. Thus, the converter generates displacement and rate error voltages to control displacement along the x coordinate. A diffraction grating with 200 lines per mm is used as a precision transducer of cutter displacement and permits accurate measurement within approximately 0.025 mm. The interference bands are sensed by two photocells, each of which delivers a pulse upon transition through a band; these pulses are counted in a counter. The direction of displacement is determined by a circuit detecting the excitation of one of the two photocells, depending on the sign of displacement. Similar control units are used for the y and z coordinates. The initial values of coordinates x, y and z are prepared with high accuracy before machining of the workpiece. The spindle axis is set up by a special dowel pin with a definite position relative to the blank workpiece. The position of the cutter is precisely fixed by a special device which determines the distance

between the cutter and the butt end of the dowel pin. The cutter position is checked before machining by inserting a pencil or scriber instead of the cutter. In the Bendix machine-tool control, the assigned discrete displacements are recorded on three tracks of the control tape for the coordinates x, y and z in straight line segments approximating the assigned profile of the article. For example, for a straight line pass of 12.5 mm at an angle of 36° 52′ to the x axis with a step of 0.005 mm and a feed rate of 500 mm/min, the x-coordinate track specifies 2000 steps and the y-coordinate track specifies 1500 steps. The "feed rate" track indicates the feed rate or execution time of the given pass, $t = 1.5$ sec. This means that for coordinate $x = 12.5 \cdot \cos 36°52 = 10\text{mm}$, 2000×0.05 pulses are to be sent to the control unit and for coordinate $y = 12.5 \cdot \cos 36°52$, 1500×0.05 pulses are to be sent. The number codes are recorded on the tape tracks in sequence, as shown in Fig. 263.

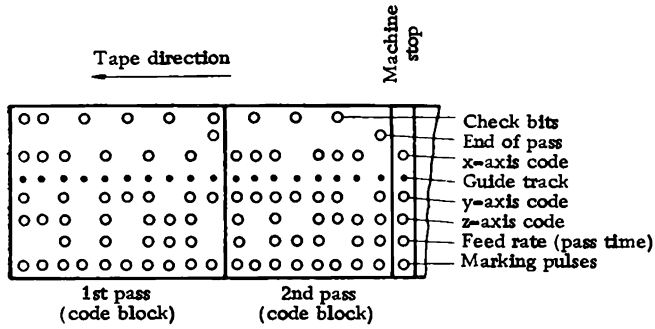


Fig. 263. Punched tape format.

The recorded code block for each pass is read and transferred to the buffer memory of the machine control unit (Fig. 264) and then transferred to the main memory, from which it is transferred to a decoder. The decoder converts the code input to equally spaced timing pulses. In the above example, during the build up time of 1.5 sec the decoder will generate 2000 equally spaced pulses to the X-axis counter and 1500 pulses to the Y-axis counter, these pulses being so synchronized that, for each four pulses arriving at the X-axis counter, three pulses arrive at the Y-axis counter. Thus, the decoder achieves uniform distribution of the pulses within the assigned time interval (i.e., it performs linear interpolation). The timing device for the control operation is a synchronized pulse generator which supplies high-frequency pulses to the decoder (and therefrom to the error counters for the coordinate axes). The operation rate can be changed by changing the frequency of the pulses. The punched tape is checked by means of a check track on which the number of recorded pulses is equal to the algebraic sum of the pulses written on tracks x, y and z divided by four. When the pulses passing through channels

x, y and z are applied to a special comparator, their sum is compared with the number of pulses written on the check track. If the number of pulses $(x+y+z)/4$ is not equal to the number of pulses on the check track, this indicates an error and the machine is stopped. The above described machine-tool control unit contains somewhat complex electrical equipment for storage of the punched-tape data for interpolation and requires considerable cabinet space (due to the use of vacuum tubes).

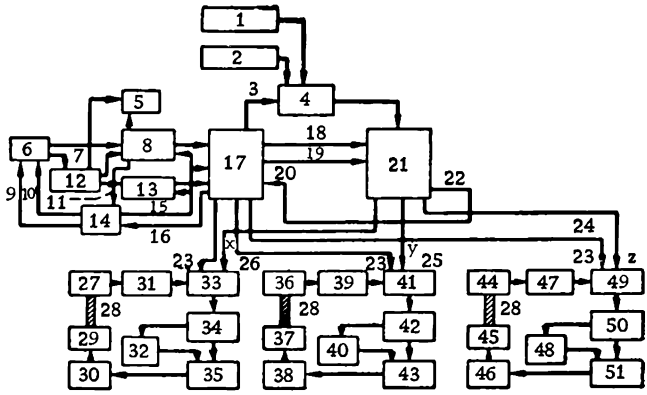


Fig. 264. Diagram of machine-tool control: 1, 2—manual control for operator; 3—feed rate; 4—auxiliary converter; 5—data check unit; 6—read unit; 7—markers; 8—temporary storage; 9—start; 10—stop; 11—end of pass (group of digits on tape); 12—read unit synchronizer; 13—block counter; 14—control of tape read; 15—clear and carry; 16—clear internal memory of machine control; 17—internal memory of machine control; 18—data for tool travel; 19—block quantity; 20—clear; 21—main data-to-pulse-sequence converter (interpolator); 22—counter overflow; 23—sign; 24—coordinate-z instruction pulses; 25—coordinate-y instruction pulses; 26—coordinate-x instruction pulses; 27—machine table pulse transducer; 28—table guide screw; 29—hydraulic driver; 30—auxiliary motor and valve; 31—demodulator; 32—auxiliary amplifier; 33—coordinate-x synchronizer; 34—x-axis comparator; 35—x amplifier; 36—pulse transducer (quantizer) of machine tool carriage for y displacement; 37—hydraulic driver; 38—auxiliary motor and valve; 39—demodulator; 40—auxiliary amplifier; 41—coordinate-y synchronizer; 42—coordinate-y comparator; 43—y amplifier; 44—pulse transducer (quantizer) of z-axis spindle head; 45—hydraulic driver; 46—auxiliary motor and valve; 47—demodulator; 48—auxiliary amplifier; 49—coordinate-z synchronizer; 50—z-axis comparator; 51—z amplifier.

The DDA may be used in preparation of the control tape since both linear and nonlinear approximation of the segments of the assigned curve profile can be computed. Computation in the DDA gives results directly in the form of unit increments of the x and y coordinates or their derivatives. These increments may be entered onto punched tape in the form of individual unit pulses. This method of preparation results in a control tape of considerable length, but the machine-control unit requires no linear approximation of segments and is thus considerably simpler. Another method (requiring a shorter punched tape) records on the tape the computed

derivatives $\dot{x}(t)$, $\dot{y}(t)$, $\dot{z}(t)$ for each straight-line segment of the profile and the time Δt for the segment. These derivatives can be obtained directly from the y registers in the DDA. The time Δt is obtained in the DDA by using a y register as a counter to accumulate unit pulses. The equidistant curve of the profile can be computed in the DDA by solving differential equations together with formulas (3) and (4). Digital programmed control of machine tools permits a sharp increase in productivity, particularly in the processing of articles of complex configuration. This is of particular interest for machine shop automation, as labor required in making the dies, forms, templates and the like can be considerably reduced. Digital programmed control of machine tools also decreases the tool wear and the setup time. Program-controlled machine tools are widely used in the aircraft industry, in machine tool industry, in the manufacture of molds and dies, and in small-lot production.

An example of machine efficiency is seen by comparing the time expended in machining a single article by manual operation which was, 215 hours, with the time required for program-controlled machining, which was 94 hours (including the setup time and the time for control tape preparation). In the latter case, the actual machine time was 5.5 hours.

3. DDA CONTROL OF MACHINE TOOLS

The computing and control capabilities of the DDA can be used for direct control of machine tools. The input information arrives at the DDA from a punched tape reader or from a ferrite-core function generator. In the milling-machine digital control system developed by the British firm Ferranti (Fig. 265a), the punched tape for insertion into the DDA is prepared by the Ferranti Pegasus computer (a digital arithmetic machine). This machine computes all the data for the DDA from mathematical formulas defining the profile of the article (for example, the profile of an air propeller or a water propeller from the given aerodynamic or hydrodynamic equations). The profile of the article may also be given by the data of a drawing entered onto punched tape from which the Pegasus computer (or a computation center where the design data are prepared) delivers the control tape for the DDA. The DDA is used to compute the increments in coordinates x, y, z and to compare the computed values with the measured values from the machine tool.

Without any control unit other than the hydraulic servo drive, the DDA directly controls the milling machine, which machines the propeller blades with a diameter of 4 feet (1.2 m). In this DDA the internal memory consists of magnetostrictive (nickel) delay lines and electromagnetic delay lines; all logic circuits are completely transistorized; the clock pulse frequency is 0.5 Mc. A machine tool with direct DDA control was demonstrated at the 1958 International Exhibit of Computing Machinery Manufacturers in London.

Let us discuss the method by which the article profile is given for computation in a DDA. Any contour of the article is a

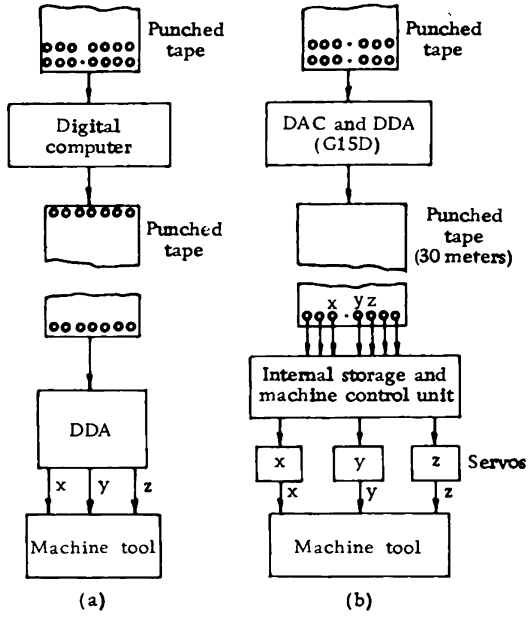


Fig. 265. Block diagrams of machine-tool control systems (a) Ferranti; (b) Bendix.

combination of segments consisting of straight lines, circles, and other curvilinear sectors which may be approximated by straight lines or parabolas. The basic data required by the DDA consists of the derivatives for three coordinates $x(t)$, $y(t)$, $z(t)$ as functions of time. For smoothly varying curvilinear contours, increments of higher-order differences instead of the derivatives are recorded on the punched tape. For example, for generation of a cubic parabola (see Fig. 266), to approximate a curvilinear segment

$$y = at^3 + bt^2 + ct + d, \tag{5}$$

the increments of the third derivative with respect to $y(t)$ are sent to the first integrator.

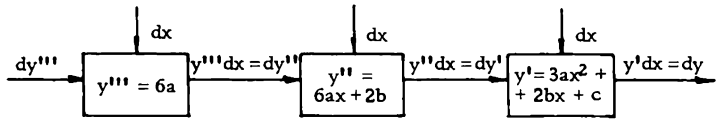


Fig. 266. Generation of cubic parabola $y = ax^3 + bx^2 + cx + d$.

Let us discuss the method of entering data into the DDA by recording the derivatives of $x(t)$, $y(t)$, $z(t)$ on punched tape. For

a straight-line segment the equation of the straight line in parametric form may be written as

$$\left.\begin{aligned} x(t) &= at\ \cos\varphi + x_0 \\ y(t) &= at\ \sin\varphi + y_0 \end{aligned}\right\} \tag{6}$$

or

$$\left.\begin{aligned} \dot{x}(t) &= a\cos\varphi \\ \dot{y}(t) &= a\sin\varphi \end{aligned}\right\} \tag{7}$$

where x_0 and y_0 are the initial values of the coordinates; $x(t)$ and $y(t)$ are the current values; a is the given feed rate (linear displacement of the cutter); φ is the angle between the straight line and the abscissa axis.

For a circular arc (Fig. 267), the equations in parametric form are

$$\left.\begin{aligned} x(t) &= x_0 + r\sin a\,t \\ y(t) &= y_0 + r(1-\cos a\,t) \end{aligned}\right\} \tag{8}$$

$$\left.\begin{aligned} \dot{x}(t) &= ra\cos a\,t \\ \dot{y}(t) &= ra\sin a\,t \end{aligned}\right\} \tag{9}$$

For an arbitrary initial position, x_0 and y_0, the center of the circle (Fig. 268) is

$$\left.\begin{aligned} x(t) &= x_0 - r\sin\varphi_0 - r\sin(\varphi_0 + a\,t) \\ y(t) &= y_0 + r\cos\varphi_0 - r\cos(\varphi_0 + a\,t) \end{aligned}\right\} \tag{10}$$

$$\left.\begin{aligned} \dot{x}(t) &= ra\cos(\varphi_0 + a\,t) \\ \dot{y}(t) &= ra\sin(\varphi_0 + a\,t) \end{aligned}\right\} \tag{11}$$

where φ_0 is the angle between the radius vector and the ordinate axis at initial position x_0 and y_0.

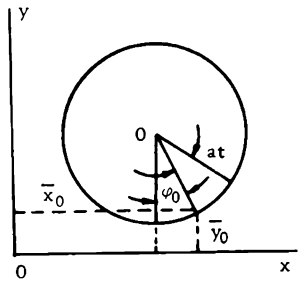


Fig. 267. Coordinates of circle with center not at coordinate origin.

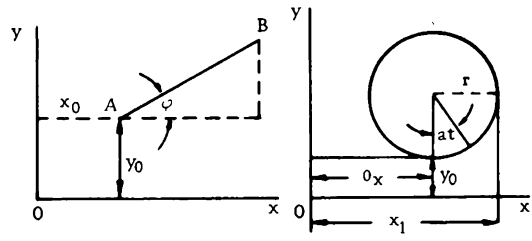


Fig. 268. Coordinates of circle with center at an arbitrary position.

In approximating a curvilinear contour as a parabolic segment, the parametric equations of this segment are

$$\left.\begin{aligned} x &= x_0 + kt \\ y &= y_0 + pt^2 \end{aligned}\right\} \tag{12}$$

$$\left.\begin{aligned} x' &= k \\ y' &= 2pt \end{aligned}\right\} \tag{13}$$

From the above equations it suffices to record on the tape the derivatives of coordinates x, y, z and the time intervals Δt for each of the approximating segments of the article contour. For example, a contour (Fig. 269) consisting of straight lines and circular arcs may be divided into six segments wherein the straight lines require $x'(t)$, $y'(t)$, $z'(t)$ and the circular arcs require, in addition, the radius of curvature r and the angle φ_0 (the initial position relative to the center of the circle).

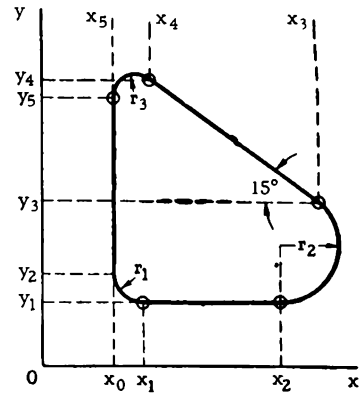


Fig. 269. An example of a profile.

Fig. 270 is a simplified block diagram showing the principle of DDA interpolation of straight-line segments (integrators 1 and 9) and parabolic segments (integrators 2, 3 and 10) as well as

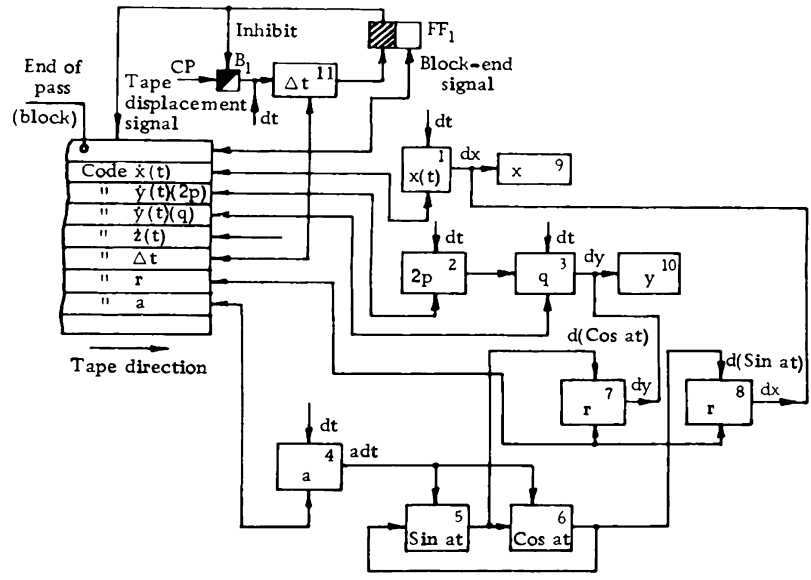


Fig. 270. Block diagram for machining an article by using a DDA.

circular arcs (integrators 4-8) from the given derivatives $x'(t)$, $y'(t)$, $z'(t)$, radius of curvature r and feed rate a. In addition, the value of the time interval Δt which is stored in the y register of integrator 11 for use as a counter must be given for each segment. Integrators 1 and 9 are used for linear approximation in the x-coordinate for both straight-line and parabolic segments. Integrators 2, 3 and 10 are used for parabolic approximation in the x-coordinate and for linear approximation when coefficient $2p$ or the term cx^2 is zero. Integrators 4-8 solve the parametric equations of a circle (for simplicity, the circle has the center located at the origin).

The computed values of dx and dy are stored in the registers of integrators 9 and 10. During computation of the points of a circle, the values of the coefficients for a parabola are recorded as zero on the tape tracks; conversely, in the parabolic approximation the radius r of the circle is recorded as zero on the tape. The timing pulses Δt arrive at the counter where the number t is accumulated; when the accumulated t reaches the assigned value of t, an overflow pulse to be stored in flipflop FF_1 appears and the counter is then cleared. Via gate B_1 flipflop FF_1 inhibits the timing pulses to the counter until the punched tape is displaced to the next block of data. The tape is moved upon the signal from FF_1. The end of the tape block is indicated on a special track. Upon a signal from this track, flipflop FF_1 returns to the initial state and the DDA performs interpolation for the next data block. Because the DDA performs interpolation of circular arcs and parabolic curves, the number of data blocks may be small. For a small number of data

blocks, the tape may be replaced by punched cards with ferrite-core circuits for reading the coded data into the computer (Fig. 271).

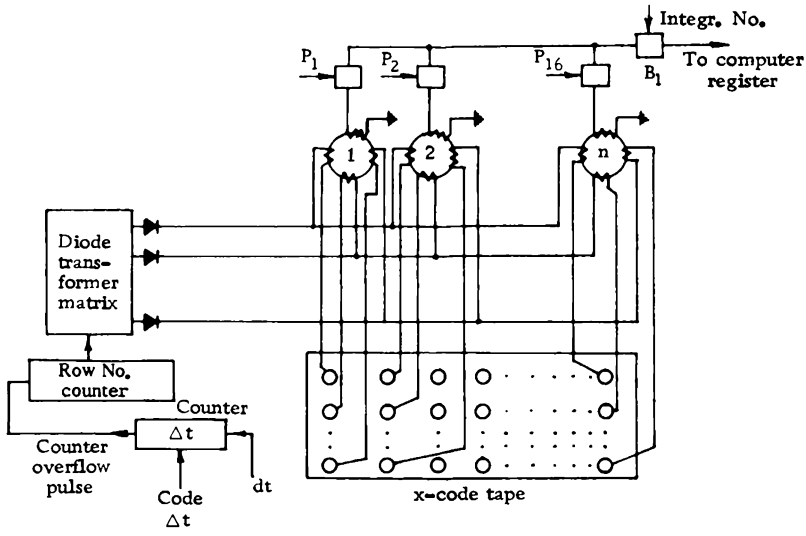


Fig. 271. Punched-card input device with ferrite-core reading circuits.

The punched card containing the number codes x' (or y',z', etc) is made of plastic. At the hole location the primary winding of a transformer is connected to the current source. Each horizontal row of holes corresponds to the one number code. The number of windings on each transformer is equal to the number of rows on the punched card. The number of transformers is equal to the number of columns on the card. Code 1 is represented in the primary and secondary windings by the presence of pulses at the CP frequency. The number is read out serially, bit by bit, by means of gates controlled by bit pulses P_1, P_2, ..., P_n of the computer. The read-out number code is transferred to the y register of the computer through gate B_1, controlled by the integrator number, upon the signal coded in the program for selection of pulses from the external channels (see Chapt. V, Sect. 9).

A set of primary windings (one winding on each core for a row) corresponding to one number code on the punched card is selected by a diode-transformer matrix which decodes the row number in the counter. The row number counter receives the overflow pulses from the Δt counter each time the number of timing pulses accumulated therein is equal to the complements of code Δt given by the punched card. The punched-card reading circuit does not require any mechanical motion of the cards; such an input device simplifies data read-out and eliminates the need of buffer registers.

Such an arrangement is considerably more convenient than the punched tape.

The advantages of using a DDA for direct machine-tool control are:

(a) One DDA may be used for control of a number of machines by multiplexing many input channels in the course of one iteration (see Chapt. V, Sect. 10);

(b) The control unit of each machine tool is considerably simplified since almost all the control functions (except for the servos, amplifiers, and transducers) are carried out in serial mode within the DDA; these include such functions as pulse counting, comparison of number codes, digital servomechanisms, entry of functions, logic operations, switching operations, and the like.

(c) The use of a DDA decreases the required amount of profile data. Instead of a punched tape with a length of scores of meters, the more reliable and stationary punched cards with electronic sensing may be used for data read-out.

(d) The DDA can control continuous motion of the tool along the prescribed profile, enable smoother displacement of the tool, and achieve a better dynamic control of the system.

4. COMBINED USE OF DDA AND DAC

A modern automatic control system is a complex aggregate of many units which must meet increasingly rigid requirements in static and dynamic accuracy, flexibility of input and output provisions, computing speed, and reliability. For this reason, the development of computers for control systems is proceeding toward the combined use of the principles of digital and analog machines. This includes the combined use of digital and analog computers for control and simulation. In this case, the exchange of information between the analog and digital computers is achieved by means of analog-to-digital and digital-to-analog converters. In these systems the digital and analog computers operate in general as independent units, although there are data exchange and synchronized operations for real-time process control.

Control system design in recent years has tended to depart from strict delimitation of digital or analog arrangement, the boundary between them gradually disappearing as the system becomes more complex and the input-output requirements become more stringent. One of these trends is the use of operational digital techniques in which the real-time control operation is achieved by a combination of individual computing elements, each performing separate algebraic operations with digital counters and other digital elements. Since these elements operate in parallel and simultaneously, the operating speed of the system is quite high. However, an important disadvantage is the complexity and inadequate flexibility of the system, rendering it suitable only for solution of one specific problem.

Another trend is seen in hybrid computers, which incorporate both digital and analog elements (e.g., digital potentiometers).

The number of these elements is determined by the functional purpose of the control system and its specific requirements. In such a hybrid computer it is difficult to distinguish the digital and analog techniques.

Hybrid systems are designed for the purpose of providing increased economy, efficiency, and reliability of the system as a whole. The most promising organization for a control system is the combination of a digital arithmetic computer (DAC) and a digital differential analyzer (DDA).

The accuracy limitation of an analog computer is avoided by the use of DDA. Real-time control applications often require stabilization of the control process by solving differential equations describing the dynamics of the system on a real-time scale with a high solution rate. With the additional requirement of high accuracy, the DDA becomes the only machine suitable for this purpose, since it is capable of a problem solution rate which is 2-3 orders of magnitude greater than the rate in a digital arithmetic machine.

Within the rather wide class of automatic control systems, the requirements for dynamic accuracy in one part of the system differ by 1-2 orders of magnitude from another part where the computation can be relatively slow or involve the processing of a large volume of data. For slow computation and data processing a digital arithmetic machine may be suitable. Computations for control purposes with high dynamic requirements are best performed by a DDA. For example, the solution rate for a serial digital arithmetic machine with a magnetic-drum memory is two complete computation cycles per second, whereas for a DDA the operating speed with the same memory is 100-200 cycles per second.

Thus, computation at a relatively slow speed and data processing of a large volume of data in a control system are performed by the arithmetic machine, while computation for process control at a high speed and with a high dynamic accuracy are assigned to the DDA. This arrangement requires interchange of information between the computers. The external channels of a DDA to a DAC are shown in Fig. 272. The results of computation performed in the arithmetic machine must be inserted as initial data into the integrand registers in the DDA. The computed values in the y registers of the DDA are cleared and replaced with new data obtained from the arithmetic machine or from external transducers. New data are entered into the DDA upon the signal generated in the arithmetic machine. The computed results in the y registers of the DDA may be used in the digital arithmetic machine; therefore, provision is made for data transfer from the DDA to the DAC or to the operator's control console upon the signal generated by both the DDA and the DAC or upon the operator's "request."

The DDA program may be altered according to the result of the problem solution. Upon the signal from the DAC, the DDA is switched from the solution of one problem to that of another with other formulas or other coefficients.

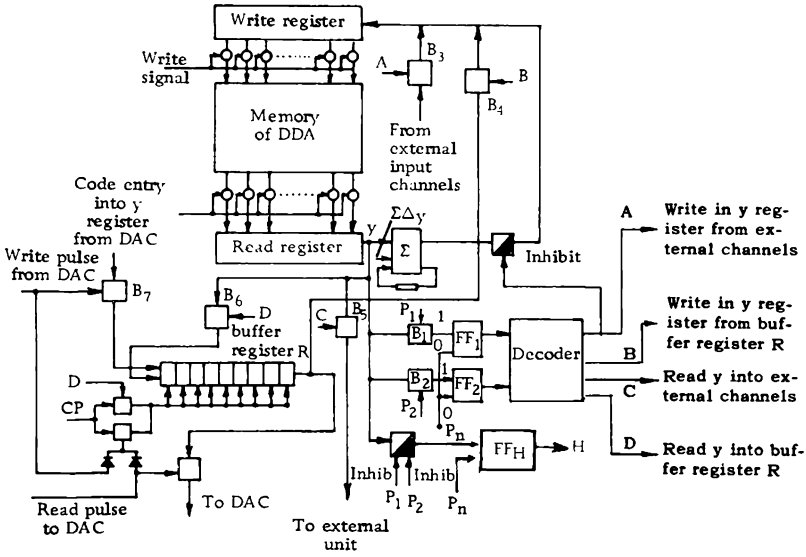


Fig. 272. Diagram of a DDA showing external channels to DAC.

Data exchange between the DAC and DDA is achieved by transferring the number codes in the y registers of the DDA to the DAC and the data from the DAC to the y registers. This operation requires the use of a buffer register R to which the data from the y register of the DDA or from the DAC are transferred.

The read and write operations in the y register (Figure 272) are controlled by the first two coded pulses in bit positions P_1 and P_2 of the number y. These two pulses pass through gates B_1 and B_2 and are stored in flipflops FF_1-FF_2, which control a decoder having four output lines A, B, C, and D. Voltage at line A unblocks gate B_3 for entry of numbers from the external channels into the y register. Line B controls gate B_4 for entry of the number code from buffer register R into the y register. Lines C and D control gates B_5 and B_6 for read-out of number y from the read register in the external channels or to the buffer register R. Transfer of the number code from the DAC to the y register of the DDA is achieved upon a DAC instruction through gate B_7 first to buffer register R and then from this register to the y register chosen by the code entered in bit positions P_1 and P_2 of this register. During this operation, the old code in the y register is replaced by a new code from the DAC. Serial transfer of several numbers from the DAC to the y registers requires control of the arrival of a number in the buffer register from the DAC until it is written into the y register. If the amount of data to be transferred from the DAC is not great, a buffer register may be provided for each number. Transfer of a number from each

buffer register to a y register in the DDA may be achieved through a gate controlled by the appropriate integrator line.

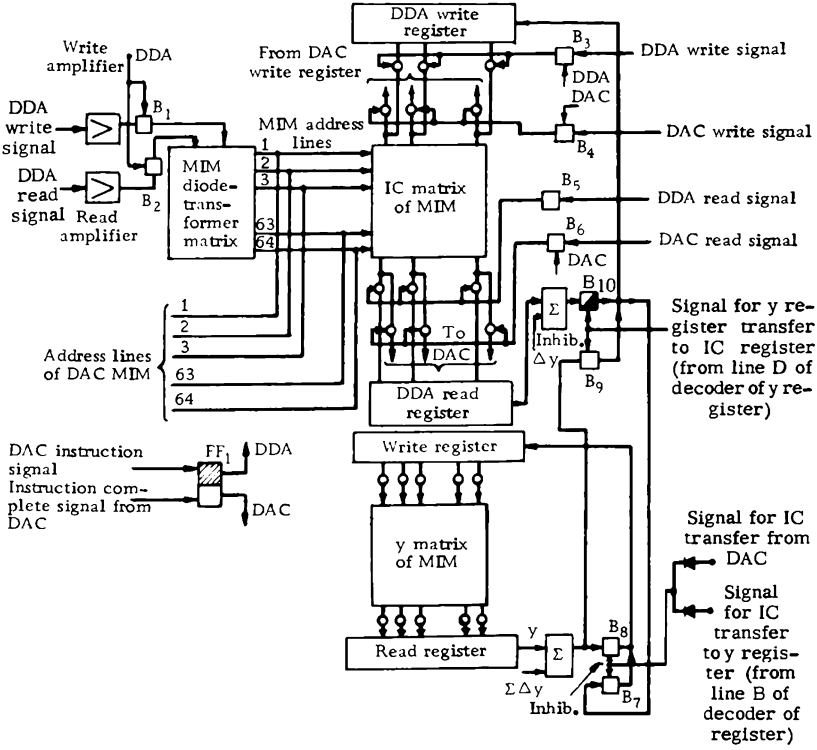

DDA write register
Write amplifier
DDA
From DAC write register
DDA write signal
DDA write signal
DDA read signal
Read amplifier
MIM diode-transformer matrix
MIM address lines
IC matrix of MIM
DAC write signal
DDA read signal
DAC read signal
Address lines of DAC MIM
To DAC
Inhib.
Signal for y register transfer to IC register (from line D of decoder of y register)
DDA read register
DAC instruction signal
Instruction complete signal from DAC
Write register
y matrix of MIM
Signal for IC transfer from DAC
Signal for IC transfer to y register (from line B of decoder of register)
Read register

Fig. 273. Exchange of data between DAC and DDA.

With a large amount of data to be interchanged between the DAC and DDA, the initial condition registers (the IC matrix) of the DDA (Fig. 273) can conveniently be used as the buffer storage. The address lines of this matrix are controlled either by the DDA diode-transformer matrix or by the lines of the DAC memory decoder. The number of address lines in the DAC is equal to the number of integrators in the DDA. The address lines from the DAC are connected in parallel to the appropriate line numbers of the diode-transformer matrix of the DDA. Thus, the memory locations of the IC matrix in the DDA can be used both as DAC memory locations and as memory locations of the IC registers in the DDA. Switching of the DDA memory locations to operate as DAC memory locations is achieved by flipflop FF_1 upon the signal from the DAC, while flipflop FF_1 is reset to the initial state upon the signal from the DAC after execution of the instruction. The

FF_1 output voltage controls gates B_1 and B_2 (switching the individual read and write amplifiers for the IC matrix in the DDA) as well as gates $B_3 - B_6$ (switching the read or write signal from the DDA or DAC). These gates, in turn, control the gates transferring the number stored in the write register of the DAC or of the DDA to the matrix, or transferring the number stored in the IC matrix to the read register of the DDA or of the DAC.

Any code stored in the IC matrix may be transferred to the y register of the DDA and, conversely, the code from any y register location may be transferred to the IC matrix by gates $B_7 - B_{10}$.

Upon a signal from line B of the y register decoder (Fig. 273), gate B_7 transfers a number from the IC matrix to the y write register. At the same time, gate B_8 inhibits the rewrite of the old code.

Upon a signal from line D of the y register decoder, gate B_9 transfers code y to the IC matrix, thereby erasing (by gate B_{10}) the old code stored in this location of the IC matrix.

Thus, a code from the y register is first transferred to the IC matrix and then to the DAC and a code from the DAC is first written in the IC matrix and then transferred to the y register.

For transfer of all codes stored in the IC matrix to the y registers, instead of a signal from line B, a signal from the DAC is applied to gates $B_7 - B_8$ at the appropriate moment, erasing all old numbers and entering the new initial values into the y register.

5. ANALOG APPLICATIONS OF THE DDA

Electronic simulation in the research and development laboratories of military and civilian technical facilities is of importance. The analog computers presently in use have a number of shortcomings:

(a) Low computing accuracy with errors of several percent. The more complex the problem is, the greater the cumulative error in computation. Beacuse of the low accuracy, the problem has to be simplified by neglecting lower-order terms; as a result, cumulative errors during integration of differential equations may yield an incorrect result.

(b) The amount of electronic equipment and the size of the analog installation increase sharply with an increase in complexity and nonlinearity of the problem. The large analog computers constructed for solving complex and nonlinear problems are extremely unwieldy.

(c) In order to obtain an accuracy of the order of even 1%, the analog computer and its elements must be accurately calibrated; such calibration must be preserved regardless of temperature changes and other environmental factors.

(d) With the long integration time associated with solution of a real-time problem, the analog computer cannot ensure even the accuracy for which it is normally designed, because of integration error due to amplifier drift and other causes.

(e) The independent variable in integration in an analog computer can only be time. This limitation deprives the analog computer of the flexibility, simplicity, and economy in generation of various functions (product, square root, quotient, etc) which can be achieved in a DDA.

(f) The number of operational amplifiers in an analog computer is considerably larger than the number of digital integrators in a DDA.

(g) The construction, testing, adjustment, and operation of operational amplifiers are more complex than of digital integrators in a DDA, since such integrators have all the advantages of digital elements over analog elements.

With the problems to be solved being more complex, the DDA becomes even a more potent tool, since in the DDA:

(a) the accuracy does not decrease with an increase in complexity of the problem;

(b) the amount of electronic equipment does not increase in proportion to the complexity of the problem;

(c) the number of required digital integrators is considerably smaller than the number of operational amplifiers.

A parallel DDA consisting of individual plugboard-connected integrators for a given problem solution has a number of advantages over analog computers, the most important of which are:

(a) The problem solution accuracy of a DDA is unattainable in analog computers. Integration errors accumulate with time in analog computers, but not in the DDA, as has been shown by tests on a DDA and by a simulation of the computing process in a DDA on the Russian "Strela" computer.

(b) The parallel DDA consists of individual standard digital integrators, which are printed circuits mounted with transistors and diodes, and a cable delay-line memory. These integrators do not require calibration and are more reliable in operation than the analog operational amplifiers.

(c) In comparison with electronic analog integrators with a capacitor as a storage device, digital integrators possess vastly greater possibilities. In the DDA, the independent variable in integration does not have to be time (as is the case in an electronic analog computer), thus permitting the use of one multiplier integrator for multiplication and division as well as for extraction of roots and raising to powers. By means of digital integrators, the DDA can generate a set of different functions of one or of several variables as well as functions given in tabular or graphical form. All these capabilities render the DDA most suitable for the solution of complex problems with a large number of nonlinear relationships. In addition, the number of integrators required for the solution of a problem in the DDA is considerably smaller than the number of operational amplifiers in analog computers, and the digital integrator is able to perform a variety of complex operations.

Bibliography

1. Tsifrovyye differentsial'nyye analizatory (Digital Differential Analyzers). A collection of translations from English. 1959. Izd. inostr. literatury.
2. MAYOROV, F. V. Tsifrovyye integriruyushchiye mashchiny. (Digital Integrating Computers). Symposium on Automatic Control and Computer Techniques. No. 1, 1958. Mashgiz.
3. MAYOROV, F. V. Primeneniye tsifrovykh integriruyushchikh mashin dlya upravleniya real'nymi ob''ektami (Real-Time Control Applications of Digital Integrating Computers). Proceedings of All-Union Conference on "Means of Developing Soviet Mathematical Machine Building," March 1956. Izdaniye Instituta nauchno-tekhnicheskoy informatsii, AN SSSR.
4. MAYOROV, F. V. Primeneniye tsifrovykh integriruyushchikh mashin dlya avotmatizatsii proizvodstvennykh protsessov (Process Control Applications of Digital Integrating Computers). Report at the Second Scientific-Technical Conference on Automatic Control and Computer Techniques (1959). Published in symposium "Avtomaticheskoye upravleniyu i vychislitel'naya tekhnika," V. V. Solodovnikova, editor. 1961. Mashgiz.
5. KLEIN, M., G. MORGAN, M. ARONSON. Digital Techniques for Computation and Control, 1958, Instruments Publ. Co., Inc., Pittsburgh, Pa. (Russian translation 1960, Izd. inostr. literatury).
6. SHILEYKO, A. V. Digital Models (a survey). Avtomatika i telemekhanika. 1960.
7. PAYSK, E. Design of Mechanisms for Generation of Functions of Several Variables. Russian translation in 'Prikladnaya mekhanika i mashinostroyeniye,' No. 2. 1953. Izd. inostr. literatury.
8. DONAN, J. E. The serial-memory digital differential analyzer. Mathematical Tables and Other Aids to Computation. Vol. 6, No. 38, April 1952, pp. 102-112.
9. SPRAGUE, R. E. Fundamental concepts of the digital differential analyzer. Ibid. Vol. 6, No. 37, June 1952, pp. 41-49.
10. MENDELSON, M. J. The decimal digital differential analyzer. Aeronautical Engineering Review. Vol. 13, February 1954, pp. 42-54.
11. PALEVSKY, M. The design of the Bendix digital differential analyzer. Proceedings of the IRE. October 1953, pp. 1352-1356.

12. BRAUN, E. L. Digital computers in continuous control systems. IRE Transactions on Electronic Computers. EC-7, no. 2, 1958, pp. 123-128.
13. BRAUN, E. L. and G. POST. Systems considerations for computers in process control. IRE Convention Record. Part 4, 1958, pp. 168-181.
14. BRAUN, E. L. A comparison of integral and incremental digital computers for process control applications. Control Engineering. Vol. 7, No. 1, 1960, pp. 113-118.
15. Avtomaticheskoye upravleniye i vychislitel'naya tekhnika (Automatic Control and Computer Techniques). Conference proceedings. 1958, pp. 392-418, Mashgiz.
16. Ekspress-informatsiya, Vychislitel'naya tekhnika (Express Information, Computer Techniques). No. 5, 1961.
17. TRICE Electronic Computer. I.R.E. National Convention Record, 1958.
18. VIII (CORSAIR). Journal Institute of Radio Engineers. 1961.
19. Communications and Electronics. 1960, pp. 393-400.
20. Poluprovodnikovaya elektronika 1959, Gosengergoizdat.
21. Dexan. World Aviation Electronics and Controls. October 1961, pp. 113-116.
22. GURAKOV, A. A. and A. G. SHEVELEV. TsDA posledovatel'nogo deystviya s mnemonicheskoy skhemoy dlya nabora zadach (Serial DDA with mnemonic scheme for problem setup). Sbornik nauchnykh trudov Kiyevskogo instituta GVF. No. I, 1961, pp. 203-215.
23. NESLUKHOVSKIY, K. S. Tsifrovyye differentsial'nyye analizatory (Digital Differential Analyzers). Moskva, 1961, Izd. Voyennoy art. akademii im. Dzerzhinskogo.
24. Kombinirovannyye vychislitel'nyye mashiny (Hybrid Computers). Proceedings of All-Union Seminar-Conference on "Theory and Methods of Mathematical Simulation" in November 1961. 1962, Akademizdat.

Supplementary Bibliography

1. BRAUN, EDWARD L. Design features of current digital analyzers, I. R. E. National Convention Record. 1954, Part 4, pp. 87-97.
2. MEYER, M. A. Digital techniques in analog systems, IRE Transactions on Electronic Computers. 1954, pp. 23-29.
3. MEISSNER, LOREN O. Real-time digital differential analyzer. Proceedings of Western Joint Computer Conference. 1954, pp. 134-9.
4. SELFRIDGE, R. G. Coding a general-purpose digital computer to operate as a differential analyzer. Proceedings of Western Joint Computer Conference. 1955, pp. 82-4.
5. HARRIS, J. N. A programmed variable-rate counter for generating the sine function. IRE Transactions on Electronic Computers. March 1956, pp. 21-26.
6. NICOLA, R. N. Operation digital techniques for special-purpose computers. Aeronautical Engineering Review. March 1956, pp. 78-82.
7. FORBES, GEORGE F. Digital differential analyzers. California, 1956.
8. KLEIN, MARTIN L., FRANK K. WILLIAMS, HARRY C. MORGAN, and SHIGERU OCHI. Digital differential analyzers. Instruments and Automation. Vol. 30, June 1957, pp. 1105-1108.
9. WEBER, H. J. Inertial guidance system uses digital integrator. Space Aeronautics. November 1958, pp. 134.
10. BECK, ROBERT M. and MAX PALEVSKY. The DDA. Instruments and Automation. Vol. 31, November 1958, pp. 1836-1837.
11. LESH, F. Methods of simulating differential analyzer on a digital computer. Journal of ACM. July 1958, pp. 291-288.
12. MITCHELL, J. M. and S. RUHMAN. The Trice—A high-speed incremental computer. IRE National Convention Record. 1958, pp. 206-216.
13. GRAY, H. J. Digital computer solution of differential equations in real time. Proceedings of the Western Joint Computer Conference. 1958, pp. 87-91.
14. GILL, ARTHUR. Systematic Scaling for Digital Differential Analyzers. IRE Transactions on Electronic Computers. December 1959, pp. 486-489.
15. VOLDER, JACK E. The CORDIC trigonometric computing technique. I.R.E. Transactions on Electronic Computers. September 1959, pp. 330-334.

16. DAGGETT, D. H. Decimal-binary conversions in CORDIC. IRE Transactions on Electronic Computers. September 1959, pp. 335-339.
17. MILAN-KAMSKI, W. J. A high-accuracy, real-time digital computer for use on continuous control systems. Proceedings of the Western Joint Computer Conference. 1959, pp. 197-201.
18. ITO, WALLY. Real-time digital analysis and error-compensating techniques. Proceedings of the Western Joint Computer Conference. 1959, pp. 269-71.
19. SKRAMSTAD, HAROLD K. A combined analog-digital differential analyzer. Proceedings of the Eastern Joint Computer Conference. 1959, pp. 94-99.
20. OWEN, P. L., M. F. PARTRIDGE, T. H. R. SIZER. The differential analyzer and its realization in digital form. Electronic Engineering. October 1960, pp. 614-7 and November 1960, pp. 700-704.
21. OWEN, P. L., M. F. PARTRIDGE and T. H. R. SIZER. CORSAIR: a digital differential analyzer. Electronic Engineering, December 1960, pp. 740-744.
22. OWEN, P. L., M. F. PARTRIDGE and T. H. R. SIZER. A transistor digital differential analyzer. Journal British IRE. August 1961, pp. 83-95.
23. DEERING, C. S. and C. B. SHELMAN. An incremental computer technique for solving coordinate-rotation equations. IRE Transactions on Electronic Computers, December 1961, pp. 748-751.
24. BRADLEY, R. E. and J. F. GENNA. Design of a one-megacycle iteration rate DDA. AFIPS. Vol. 21, 1962, pp. 353-364.
25. CONNELLY, MARK E. Real-time analog-digital computation. IRE Transactions on Electronic Computers. February 1962, pp. 31-41.
26. TRUITT, T. D. An analog-digital real-time computer. IRE Transactions on Electronic Computers. February 1962, pp. 46-52.
27. PALEVSKY, MAX and J. V. HOWELL. Digital differential equation solver. Instruments and Control Systems. April 1963, pp. 118-121.
28. EDGE, C. F. Digital differential analyzers vs. general-purpose digital computers for Schuler-tuned Intertial Navigation Systems. IEEE Transactions on Military Electronics. January 1963, pp. 23-29.

Index

Acoustic delay lines. See Delay lines
Adders
 decimal, excess-3-code 268–72
  four-bit 5211 code 264–68
  increment sum 272–74
 in multiplier integrator 335–39
 increment 339–41
 serial binary 166
 single-bit 167, 168, 169
 three-input 166
Addition 57
 of decimal numbers 264
Address current driver 191
Address decoding 183–87
 diagram for complete unit 187
Address lines
 drivers 191–95
 in magnetic internal memory 181
 selection of 344
 selection switch 191
 switching 252
Address tracks 32, 34
Addressing circuits 40
Algebraic equations. See under specific type
Amplifiers
 inverter. See Inverter amplifier
 magnetic drum read 175
 magnetic drum storage read-write 174–76
 magnetic drum write 176
 null, in multichannel converter 286–87, 290
 power, for CP and CPS generators 156
  two-stage 112
  with cascaded transistors 113
 read 198–203
  for magnetostrictive delay lines 210
 read-write, for magnetic drum storage 174
 two-stage, in dynamic flipflop 163
 write 197
  for magnetostrictive delay lines 210
Analog computer 352–53
 disadvantages of 366, 369
Analog-to-digital converters 275–300
AND circuits
 in adders 166
 with emitter followers 111–12
AND coincidence circuit 141
AND gate in dynamic flipflop 147
Arc-cosecant 68
Arc-cosine 67
Arc-secant 68
Arc-sine 66
Arc-tangent 67
Automatic control of production processes 348–53

Bessel equation 69
Bessel functions 69
Binary code, nonreturn-to-zero 327
Binary coding 14–18
 binary system 15
 decimal system 18
Bistable multivibrator with diode limiters 139

Cable, coaxial, as delay line 325, 326–29
Cartesian coordinates
 transformation of spherical coordinates to 73–75
 transformation to polar 75–76
Clock pulse generators 155–57
Clock pulse shapers 155–57
Clock pulse track 31
Clock pulses 32, 35, 37, 40, 101
 see also Pulse
Code converter, decimal digit 268
Code disk with brushes 297–300
Codes
 binary. See Binary
 decimal. See Decimal
 excess-3 268–72
 5211 264–68
Coding, ternary method 14, 23–30
Coincidence circuits 56
 diode 140
 diode-transformer 142, 151, 154, 163, 170–71, 189, 200, 205
 gate amplifier 144
 parallel integrating computer 329
 pulse storage 144
Complex functions 71

Computers
 CRC-105 2
 Mars 1000 350
 see also Analog computer; Digital arithmetic computer; Digital differential analyzer; Digital integrating computer; Parallel digital integrating computer; Serial binary digital integrating computer; Serial decimal digital integrating computer
Control, automatic, of production processes 348–53
Control unit 222, 231–33
Converters
 block diagram 294
 coding disk 299
 control unit 293
 digital potentiometer 284
 multichannel, using digital potentiometer 286
 multichannel analog-to-digital 275–300
 multichannel units 276
 shaft-position 297–300
 using diode switches 276
 using transistor switches 281
 voltage 275
 voltage, ten-bit 279
Coordinate transformation 72–73
Correlation analysis 98–100
Correlation function 99
Cos x 64
Cosecant 65
Cosecant squared 66
Cotangent 65, 66
Cotangent squared 66
Counters
 integrator 37
 integrator number 234
 reversible 160
 ring 159
  for MIM diode-transformer matrix 184
  in decorder 163
  in integrating unit 234
  one-bit position 161
  pulse time diagram 160
  reversible decimal-digit 273
 three-digit 35
Counting circuits 159–63
CRC-105 computer 2
Cross coupling 108, 115, 134–35
Curve follower 308–09

Decimal adder
 excess-3-code 268–72
 four-bit 5211 code 264–68
 increment sum 272–74
Decimal codes 265
Decimal DDA. See Digital differential analyzer
Decimal digit code converter 268
Decimal number system 18, 259
Decimal numbers, addition and subtraction 264
Decision integrator, servo integrator as 48
Decoders
 diode, in function generator 315
 diode-transformer 163
  in function generator 315
 for six-bit number code 165
Decoding, address. See Address decoding
Delay element 126
Delay lines
 acoustic 213–19
  access time with 214
  control elements 217
  pulse modulation 213
  read, write and erase circuits 219
  temperature control 214
 as dynamic flipflop storage element 152–53
 coaxial cable as 325, 326–29
 electromagnetic 253
 for storage of current overflows 240–42
 magnetic-drum 33
 magnetostrictive 206–13
  as standard storage registers 212
  characteristics of 208
  electronic components associated with 210
  in decimal DDA 264
  parallel connection in DDA 211
  read and write amplifiers 210
  temperature compensation 211
 ultrasonic 252–56
  advantages and disadvantages 255–56
  characteristics of 255
Derivatives, lowering the order of 54
Differential equations
 solution of 54–57
 with automatic change of coefficients 92
 with given boundary (initial and final) conditions 80–82
Differentiation 82–86
 approximate 84
 correction circuit 84, 85
 drift signal 86
 error signal 84, 85, 86
 of graphically given function 85
Digit number 101

Digital arithmetic computer 350, 351
 combined use with digital differential analyzer 365–69
Digital differential analyzer 350–52
 advantages of 2, 3, 4
 analog applications 369
 applications 348–70
 basic elements 4
 basic operations performed 3–4
 basis of operation 1
 combined use with digital arithmetic computer 365–69
 comparison of characteristics with VIDIC and DAC 6
 CRC–105 2
 disadvantages 5
 incremental 5
  advantages of 6
  operations performed by 5
 machine-tool control by 353–59, 359–65
 parallel type 2
 serial 1
  advantages and disadvantages 4
  coded programme for 3
 types 1
 variable increments 5
 with magnetic drum 259
Digital integrating computer. See Digital differential analyzer
Digital integrator 1, 329–34
 block diagram 7
 comparison with wheel-and-disk integrator 8
 errors of 10
 logic diagram 331
 operation of 9
 trapezoidal method 330–31
Digital potentiometer 284
 coarse stage 288
 fine stage 289
 multichannel converter using 286
Digital servo 341–44
Diode capacitor read register 203
Diode-capacitor registers 195–97
Diode coincidence and inhibit circuits 140
Diode decoder. See Decoders
Diode flipflop 151
Diode switches, converters using 276
Diode-transformer coincidence and inhibit circuits 142
Diode-transformer coincidence circuit 151, 154, 163, 170–71, 189, 200, 205
Diode-transformer decoder. See Decoders
Diode-transformer matrix 43, 44, 317, 319, 321–22, 344
 magnetic internal memory 183
Diodes, silicon 136, 138–40
Divider resistances of flipflop circuit 107
Division 59
Double integrator 257
Drift signal 86
Drivers
 address current 191–92
 address line 191–95
 quartz 219
 read address 191, 192
 write address 191, 195
Dynamic flipflops 147–53
 delay line as storage element 152–53
 false triggering of 154
 for increment adder 341
 for MIM diode-transformer matrix 184
 high-power 153–55
 in decimal adder 273
 in digital integrator 331–32
 in digital potentiometer 285
 in diode-transformer decoder 163–66
 in function generator 322–23
 in overflow storage line 249
 in ring counter 159, 161
 in voltage converter 281
 operation of 155
 positive differentiating feedback in 149
 shift registers 157–59
 two-stage amplifier for 163
 two-transistor 151
 with diode elements 151
 with storage capacitor 147
 with storage in transistor 151
 with voltage level output 150

Elliptic functions 70
Emitter followers 109–15, 127
 AND circuit with 111–12
 as intermediate amplifiers 121
 at flipflop input and output 121
 at flipflop output 120–21
 base current 109
 bias voltage 110
 collector voltage 110
 combined with inverter amplifier 112
 dynamic flipflop 150
 emitter current 109
 gain of 110
 in inhibit circuit 145
 input resistance 109–10

Emitter followers (continued)
OR circuit with 111
output resistance 110
output voltage 110
parallel connection of 111
voltage gain 110
with n-p-n transistor 110
with p-n-p transistor 110
Equations. See under specific type
Error signal 84, 85, 86
Excess-3-code decimal adder 268-72
Exclusive-OR circuit 132-33
Exponential function 60
Extreme values 86-88

Feedback in dynamic flipflop 149
Feedback loop 84
Ferrite-core memory for initial data storage 344
Ferrite-core transformers in function generators 315, 317, 319
Ferrite cores 187-91, 205
in write amplifier 198
Ferrite plates 176, 179, 181, 182, 186
5211-code decimal adder 264-68
Flipflops 31, 35, 36, 37, 40
block diagram 128
divider resistances of circuit 107
dynamic. See Dynamic flipflops
faster switching of 118, 120-21
formed by two current switches 134-35
in function generator 311
in graphical plotter 313-14
in logical sign operation 94
inverter amplifiers in circuit 128
semiconductor 107
stable state of 121
stable states of circuit 104
static 108
static transistor 115-24
switching from one stable state to another 115
symmetrical static 115
transformer-coupled 118
unsymmetrical static, with emitter coupling 124-26
vacuum-tube 107
waveforms 118, 120, 123, 124
with common input 115-16
with cross-coupled inverters 128
with diode limiters 138
with emitter followers at input and output 121
at output 120-21
with inductances 116
Fredholm-Volterra integral equation 95
Function $dz = -\frac{dU}{1+V}$ 63
$dz = -\frac{dx}{y}$ 63
Function generators 55, 99
as input device 308
ferrite-core 315-16
for pulse-time code for telemetry 319
high-speed 314-23
reading of increment pulses from 310-11
servo-driven 308-09
two-variable 321
Function limiter 88-91

Gate amplifier 145
Gauss-Siedel method for integral equations 95
Graphical data
conversion into punched-tape codes 312
input devices for 308-13
Graphical plotting of computed results 313-14

Higher-order algebraic equations, solution of 78-79
Hyperbolic functions 69

Impulse transfer function 100
Increment adder 339-41
Increment coding, ternary system 14
Increment input unit 236
Increment-pulse storage 240-42
Increment pulses
from function generator 310-11
writing and reading 242
Increment selections from storage lines 245
Increment storage lines 242
Increment sums
for decimal adder 272-74
positive and negative values 274
Increments
coding of 23
ternary method 23-30
Inhibit circuits
diode 140
diode-transformer 142
pulse storage 144
with emitter follower 145
with inhibit pulse applied to emitter 146
Initial condition register 256

Initial conditions
 insertion of 41, 43
 selection of 44
Initial data insertion 344–47
Initial solution time 42
Input data
 parallel integrating computer 324
 read-write operation 251
 storage in MIM 251
Input devices
 for graphical data 308–13
 function generator 308
 punched tape 301–08
Input-output devices 301–23
Input parameters 3
Input unit, increment 236
Integral
 accumulation of 19
 approximate representation of 7
Integral equations 95–98
 block diagram for 98
 Fredholm-Volterra 95
 iteration method 95
Integral sine function 69
Integrating unit 224–31
 parallel integrating computer 324
 with ultrasonic delay lines 253–55
Integration
 coding of increments 23
 initial solution time 42
 insertion of initial conditions 41, 43
 "mean" rectangle method 12
 operation of 19
  with magnetic-drum memory 34
 principles of 7
 trapezoidal method 11, 12–14, 330–31
Integrator counter 37
Integrator number counter 234
Integrators
 decision, servo integrator as 48
 digital 1, 329–34
  block diagram 7
  comparison with wheel-and-disk 8
  errors of 10
  logic diagram 331
  operation of 9
  trapezoidal method 330–31
 double 257
 interconnected, scale factors for 51
 interconnection of 38–40
 multiplier 334–39
 scaling 52, 58, 93
 servo. See Servo integrators
 single, scale factor for 50
 summing 48, 57–58
 wheel-and-disk 8
Inverse function 61
 square of 61
Inverter amplifier 107–09, 126, 127
 combined with emitter follower 112
 cross coupling 108, 115
 in flipflop circuit 128
 in transistor logic circuits 109
 with several inputs 108
 with single input 108
Ionization 136
Iteration methods for integral equations 95

Jacobean elliptic function 70
Junction transistors
 amplitude of output voltage pulse at collector 107
 base current 103–04, 106
 base-emitter voltage 104
 collector characteristic 103
 collector current 102, 106
 current gain 102
 cutoff 102
 cutoff region 103
 emitter current 102
 high-frequency 109
 initial collector current 102–03, 106
 linear region 103
 operating regions 103
 saturated region 103–04
 switching operation 102–07
 see also Transistors

Leakage current 283–84
Limiting operation 88–91
 block diagram for 89
 with binary code 90
Linear first-order algebraic equations, solution of 76–77
Logarithmic functions 61

Machine tools
 control by digital differential analyzer 359–65
 digital programmed control 353–59
Magnetic drum, decimal DDA with 259
Magnetic-drum memory 31–38
 address tracks 32
 clock pulse track 31
 clock pulses 32
 disadvantages of 252–53
 integration operations 34
 overflow line 38
 S tracks 31
 ΔS tracks 32
 tracks being delay lines 33
 y tracks 31

Magnetic drum storage 172
 dynamic recording 174
 nonreturn-to-zero recording 174
 read-write amplifiers 174–76
 read-write operation 172–74
Magnetic internal memory 176–83
 address lines, switching 252
 block diagram 177
 coincident current selection 187
 diode-transformer matrix 183
 ferrite cores 187–91, 205
 ferrite plate 179, 186
  advantages 182
  read and write lines 181
 ferrite plates or cores 176
 "forced" read-out 183
 in serial binary digital integrating computer 222
 input channel data storage 251
 linear selection 187
 matrix diagram 187
 read-write operation 178, 187–91
 read-write time cycle 178
 registers 195–97
 storage of polynomial coefficients 317
 summing of magnetic fluxes in read-write operations 181
 without half-select current coincidence 205
Magnetostrictive delay lines. See Delay lines
Mars 1000 computer 350
Multichannel analog-to-digital converters 275–300
Multichannel transistor switch 295
Multiplication 58, 334
 examples 335
 rules for 334
Multiplier integrator 334–39
Multivibrator
 bistable, with diode limiters 139
 one-shot 261

Nonreturn-to-zero binary representation 327
NOR circuits 130–31
 waveforms of voltage pulses 132
 with n-p-n transistors 134
NOT circuits, in adders 166
NOT operation, notation for 133
Null amplifier in multichannel converter 286–87, 290
Number coding 14–18

Operation integrator, servo integrator as 48
OR circuits 130–31
 in adders 166
OR circuits (continued)
 waveforms of voltage pulses 131–32
 with emitter followers 111
 with n-p-n transistors 134
Oscillations, damping of 84
Output converter for motor control 238–40
Output data, parallel integrating computer 325
Overflow line 38
Overflow pulse codes 19–23, 40
Overflow pulse sequences 20–23
Overflow pulse storage 240–42, 253
 dynamic flipflops 249
Overflow pulses 9–11, 34, 36–38, 45, 48, 49, 52, 53, 93, 94
 selection of 40, 246
 status of 39
Overflow storage lines 101
 shift registers as 246
 using flipflops 249

Parallel digital integrating computer 324–47
 block diagram 324–26
 data input 324
 delay-line storage register 326
 digital integrator 329
 digital servo 341
 initial data insertion 344
 integrating unit 324
 multiplier integrator 334
 output data 325
 pulse generators 326
 pulse increment adder 339
 units and their functions 324
Plotter for graphical recording of computed results 313–14
Polar coordinates
 transformation of Cartesian coordinates to 75–76
 transformation to Cartesian 73–75
Polynomial of nth degree 71
Polynomial coefficients
 approximating to curve segments 317
 storage in MIN 317
Potentiometer
 digital 284
  coarse stage 288
  fine stage 289
  multichannel converter using 286
 transducer, in multichannel converter 286
Power amplifier
 for CP and CPS generators 156
 two-stage 112
 with cascaded transistors 113

Power function 63
Probability function (probability integral) 62
Problem switching 41
Production processes, automatic control 348–53
Program code generation unit 233
Program preparation 100–01
Pulse, overflow. See Overflow pulses
 see also Clock pulse
Pulse generator, parallel integrating computer 326
Pulse increment adder 339–41
Pulse modulation 213
Pulse number 101
Pulse storage line, one-cycle 249, 250
Pulse transducer 311
 for graphical plotter 313
Punched card program unit 233
Punched cards for machine-tool control 364
Punched tape
 conversion of graphical data for 312
 for machine-tool programmed control 353–55, 357
 input device 301–08

Read amplifier 198–203
 for magnetostrictive delay lines 210
Read register, diode-capacitor 203
Read-write operation
 in cable delay line 327
 in magnetic internal memory 178, 181, 187–91
 initial data 344–47
Reference pulse 37
Registers
 diode-capacitor 195–97
 initial condition 256
 intermediate 195, 196
 read, diode-capacitor 203
 series-to-parallel converter 195
 storage, for initial data insertion 344
Relay switches in digital potentiometer 284
Right triangle, solution of 72–73

Scale factors 49–53, 100
 change in 58
 for interconnected integrators 51
 for single integrator 50
 multiplication diagram 52
Scaling integrator 52, 58, 93
Schmitt trigger 124
Secant 65
Secant squared 65
Serial binary digital integrating computer 220–58
 block diagram 220
 control unit 222, 231–33
 double integrator 257
 function input unit 220
 general characteristics 223
 increment input unit 236
 increment-pulse storage 240–42
 increment pulse writing and reading 242
 initial condition register 256
 input and conversion unit 220
 input data 220
 integrating unit 224–31
 magnetic-core internal memory 222
 output converter for motor control 238–40
 output unit 223
 program code generation unit 233
 storage of input channel data 251
 timing diagram 243
 ultrasonic delay lines 252–56
Serial decimal digital integrating computer 259–74
Servo, digital. See Digital servo
Servo integrator 17, 44, 57–59, 77, 83, 84, 87, 96, 332
 as function limiter 88–91
 as operation (or decision) integrator 48
 as summing integrator 48
 comparison of two quantities by 47
 example of use of 48
 operating principle 45
Shaft-position converter 297–300
Shift registers 34, 35
 as overflow storage lines 246
 for increment adder 341
 high-speed flipflop 157–59
Sign-alternating function 91
Sign operation 91–94
Silicon diodes 136, 138–40
Sin x 64
Speed of problem solution 2, 3
Spherical coordinates. See Polar coordinates
Square of inverse function 61
Square root 60
Squaring 60
Statistical analysis 98
Storage capacitors 196, 197, 200
Storage devices 172
Storage elements, diode capacitor 203
Storage registers, for initial data insertion 344
Subtraction 57
 of decimal numbers 264

Summing integrator 57–58
 servo integrator as 48
Switches
 address line selection 191
 control of several switching circuits 130
 diode, converters using 276
 flipflop formed by 134–35
 high-speed current, using n-p-n transistors 129
  using p-n-p transistors 128
 relay, in digital potentiometer 284
 transistor 134
  converter using 281
  in digital potentiometer 284
  multichannel 295
 unipolar, transistors as 290
Switching
 control of several circuits 130
 of address lines 252
 operation of junction transistor 102–07

Tangent 65
Tangent squared 65
Ternary method of coding, 14,23–30
Threshold voltage 136
Timing diagram 243
Transducer potentiometers in multichannel converter 286
 see also Pulse transducer
Transformation
 coordinate 72–73
 of Cartesian coordinates to polar 75–76
 of spherical (polar) coordinates to Cartesian 73
Transformer-coupled flipflops 118
Transformers, ferrite-core, in function generators 315,317,319
Transistor, junction. See Junction transistors
Transistor switches
 converter using 281
 in digital potentiometer 284
 multichannel 295
Transistors
 as unipolar switches 290
 characteristics of P11 and P16A 290
 see also Junction transistors
Trapezoidal integration 11, 12–14, 330–31
Trigonometric functions 64–68

Ultrasonic delay lines. See Delay lines

Voltage converter 275
 ten-bit 279
Voltage digitizers 275
Voltage divider
 in multichannel conveter 286
 unipolar switches in 290
Voltage limiters 136
Voltage regulation factor 137
Voltage regulators 136

Wheel-and-disk integrator 8
Write. See Read-write operation
Write amplifier 197–98
 for magnetostrictive delay lines 210

Zener current 138
Zener effect 136